Decision Procedures for the Security of Protocols with Probabilistic Encryption against Offline Dictionary Attacks

Stéphanie Delaune, Florent Jacquemard

Decision Procedures for the Security of Protocols with Probabilistic Encryption against Offline Dictionary Attacks *

Stéphanie Delaune† (stephanie.delaune@lsv.ens-cachan.fr)
LSV, ENS de Cachan & CNRS, France Télécom R&D

Florent Jacquemard (florent.jacquemard@inria.fr)
LSV, ENS de Cachan & CNRS, INRIA Futurs, Project SECSI

Abstract. We consider the problem of formal automatic verification of cryptographic protocols when some data, like poorly chosen passwords, can be guessed by dictionary attacks. First, we define a theory of these attacks and propose an inference system modeling the deduction capabilities of an intruder. This system extends a set of well studied deduction rules for symmetric and public key encryption often called Dolev–Yao rules with the introduction of a probabilistic encryption operator and guessing abilities for the intruder. Then, we show that the intruder deduction problem in this extended model is decidable in PTIME. The proof is based on a locality lemma for our inference system. This first result yields to an NP decision procedure for the protocol insecurity problem in presence of a passive intruder. In the active case, the same problem is proved to be NP–complete: we give a procedure for simultaneously solving symbolic constraints with variables which represent intruder deductions. We illustrate the procedure with examples of published protocols and compare our model to other recent formal definitions of dictionary attacks.

Keywords: Verification, Cryptographic protocols, Formal methods, Dictionary attacks, Probabilistic encryption.

1. Introduction

The formal verification of cryptographic protocols in an insecure network is known to be an undecidable problem, even with strong restrictions. The undecidability results from several factors: the ability of agents to generate fresh random data (nonces), the unlimited size of terms, the unboundedness of the number of sessions. Removing the last condition is sufficient for decidability (while removing the others is not, see (Durgin et al., 1999; Comon and Cortier, 2005; Amadio and Charatonik, 2002)) and several procedures have been proposed to decide the

* This work has been partly supported by the RNTL project PROUVÉ 03V360 and the ACI-SI Rossignol.

† Corresponding author: Stéphanie Delaune, LSV, ENS de Cachan & CNRS, 61 avenue du Président Wilson, 94235 Cachan Cedex - France *tel:* +33 1 47 40 75 32 *fax:* +33 1 47 40 75 21 (delaune@lsv.ens-cachan.fr)

protocol insecurity problem with a bounded number of sessions (Amadio and Lugiez, 2000; Millen and Shmatikov, 2001) and (Rusinowitch and Turuani, 2001) where the problem is shown NP–complete.

In the works cited above, as well as in many other approaches concerning automated verification of security protocols, the intruder model is based on the so–called Dolev–Yao deduction rules for public–key encryption (Dolev and Yao, 1983). This deduction system specifies how the attacker can obtain new information from previous knowledge, which he has obtained by silently eavesdropping the communication network (it is a *passive* intruder behavior) and by participating to protocol by sending new messages, thus provoking honest participants to reply according to the protocol rules (in this latter case, the intruder has an *active* behavior). The deduction abilities of the intruder still comply to the *perfect cryptography assumption*, which states that the only way to obtain knowledge about an encrypted plaintext is to know the decryption key, and there is no way to obtain the key from ciphertexts. This abstraction happened to be accurate enough to reveal many logical attacks on known cryptographic protocols in an automated way. However, it may be too strong to capture some specific attacks that may occur in real word situations. For instance, in the Dolev–Yao model it is not possible to take into account attacks based on algebraic properties of cryptographic operators, like exclusive or, in presence of which the protocol insecurity problem with a bounded number of sessions is still decidable (Comon-Lundh and Shmatikov, 2003; Chevalier et al., 2003).

In this paper, we formalize another interesting attack technique which appears to be out of the scope of the Dolev–Yao model: the so–called *dictionary attacks* (Gong et al., 1993; Lowe, 2004). In some situations, an intruder is able to guess poorly chosen passwords (or other data belonging to a reasonably small domain) by an *offline* brute force iteration through a *dictionary*, using messages previously collected on the network to verify his guess at each step. The reason for the interest of this kind of attacks is simple: password–guessing attacks are a common avenue for breaking into systems, and the application of formal method to analyze password protocols can help.

There are number of facets to the study of dictionary attacks. At the beginning of the nineties, several examples of dictionary attacks have been analyzed, and some countermeasures have been proposed to design protocols resistant to this kind of attacks (Gong et al., 1993; Gong, 1995; Tsudik and Herreweghen, 1993; Bellovin and Merritt, 1992). Lot of effort seems to have been put into using *provable security* (Goldwasser and Micali, 1984) for analysis such protocols that use poorly–chosen data (Wu, 1998; Bellare et al., 2000; Katz et al., 2001). However, the automatic verification methods which have been used successfully for cryptographic protocol analysis, were not used at this time for password protocols. Perhaps this is due to the complex nature of dictionary attacks, whose analysis involves complications similar to combining cryptanalytical and abstract protocol analysis. Only recently, some procedures have been implemented to automatically find dictionary attack (Lowe, 2004; Corin et al., 2003; Cohen, 2002; Blanchet, 2004). However, neither the complexity or the completeness of the procedures, nor the decidability of the problem have been studied in these works.

We propose decision procedure (and complexity) results for the decision of the security of protocols against dictionary attacks, when the number of sessions is bounded. Our model of dictionary attacks is based on an inference system extending the Dolev–Yao rules for symmetric and public–key encryption to take into account probabilistic encryption and offline dictionary attacks. Our inference rules for dictionary attacks are semantically closed to the definition of (Lowe, 2004). The first result is a locality result showing that the intruder deduction problem is PTIME. The second result concerns the case where the intruder is active. We show that the problem is NP–complete by reducing it to the problem of the satisfaisability of symbolic constraint systems. Though this complexity is the same as in the Dolev–Yao model, see (Rusinowitch and Turuani, 2001), the proofs of our decision procedure are made dramatically harder by the introduction of guessing abilities in the inference system. Indeed, some basic results easy to prove in the standard Dolev–Yao model are not basic anymore in our extended model.

Moreover, the extension of the intruder deduction model to probabilistic encryption operators is an important contribution when considering vulnerability to dictionary attacks. Indeed, this kind of encryption can be used as a countermeasure to prevent dictionary attacks.

After some motivating examples of dictionary attacks (Section 2) and preliminary definitions of protocols syntax and semantics (Section 3) we define in Section 4 our extended intruder model which formalize in particular dictionary attacks and we give in Section 5 decision problems (intruder deduction, trace insecurity, protocol insecurity) we are interested in. We then prove a locality result from which it follows that the intruder deduction problem protocol can be decided in polynomial time (Section 6). This yields to an NP algorithm for the trace and protocol insecurity problems in presence of a passive intruder. In Section 7, we reduce the trace and protocol insecurity problems in presence of an active intruder to the satisfaisability of symbolic constraint systems (where each individual constraint is a lifting of the problem defined in the passive case). We give in Section 8 a non–deterministic polynomial time procedure to decide the satisfaisability of such constraint systems. A comparison with other models based on

CSP (Lowe, 2004) and the applied pi–calculus (Corin et al., 2004) can be found at the end of the paper.

A preliminary version of this work as been published in (Delaune and Jacquemard, 2004).

2. Examples

A simple subcase of dictionary attacks is the *known–plaintext* attacks, where an intruder intercepts a message $\{M\}_K$ encrypted with a weak password K and whose encrypted content M is known (for instance an instruction like `hello`). The intruder can then try to decrypt this ciphertext with each word in a dictionary one by one, and verify for each guess d whether the value obtained is the known plaintext M, which means with a high probability that $d = K$. This method also works against a challenge–response scheme where a server sends to a user A a random number (*nonce*) N as a challenge and A responds with $\{N\}_K$, where K is its weak password.

The examples below show that similar attacks are also possible in some cases where the plaintext is not known, with more subtle techniques to verify the guesses.

2.1. Naive Vote Protocol and Probabilistic Encryption

Consider the following naive vote protocol:

$$0.\ A \rightarrow S : \{V\}_{pub(S)}$$

The voter A encrypts his vote V with the public key $pub(S)$ of the vote server S. The server decrypts the message with his private key $priv(S)$ and registers the vote. The security requirement is that only A and S know V. This protocol is secure if we assume strong encryption primitives: an intruder who intercepts the message $\{V\}_{pub(S)}$ will not be able to learn the vote V of A as long as he does not know $priv(S)$. However, if we assume that the intruder knows a finite set $\mathcal{D}$ (reasonably small) of values that V can take, then he can deduce V without knowing $priv(S)$: for each value $d \in \mathcal{D}$, he encrypts d with $pub(S)$ and verifies whether the ciphertext $\{d\}_{pub(S)}$ obtained is equal to $\{V\}_{pub(S)}$, which means that the guess $d = V$. Therefore, an intruder able to eavesdrop the vote messages will be able to deduce who voted for whom.

The attack described above can be mounted only if the encryption scheme is deterministic. This means that encrypting a given message with a key always returns the same result. This is the case of encryption algorithms such as *e.g.* RSA and AES. To prevent such of failure, in real vote protocols, the algorithms used for encryption often include random choices and therefore are probabilistic. This kind of encryption scheme was invented by Shafi Goldwasser and Silvio Miscali (Goldwasser and Micali, 1984) and recent practical implementations have been proposed. The idea behind probabilistic encryption is roughly that the encryption algorithm relies on some random value, chosen by the agent involving encryption. It means that encrypting several times the same plaintext with the same key returns different ciphertexts. However, the decryption function will return the same plaintext from all these ciphertexts.

Let us consider the naive vote protocol is based on probabilistic encryption. We model this situation by adding a third parameter to the encryption primitive, which represents the random choice performed by the agent before applying the encryption algorithm. Hence, we obtain:

$$0.\ A \rightarrow S :\ \{V\}^{r}_{pub(S)}$$

Assume that the attacker intercepts the ciphertext C (*i.e.* $\{V\}^{r}_{pub(S)}$). Even if he guesses V correctly, encrypting V with the public key of S, the result will be a completely different ciphertext C'. He, therefore, cannot compare C and C', and so cannot know that he has guessed the message V correctly.

2.2. Handshake Protocol

Consider this challenge–response transaction which is commonly used in authentication protocols, see (Gong et al., 1993):

$$\begin{array}{l} 0.\ A \rightarrow B :\ \{N\}^{ra}_{pw(A,B)} \\ 1.\ B \rightarrow A :\ \{N+1\}^{rb}_{pw(A,B)} \end{array}$$

A generates a nonce N and sends it to B encrypted probabilistically with the symmetric key $pw(A,B)$ (the cryptosystem is symmetric in this example). B decrypts the message, computes $N+1$, and returns to A the encrypted result. In the standard Dolev–Yao model, an intruder who intercepts the messages cannot deduce $pw(A,B)$, and the incrementation of N in the second message prevents replay attacks. However, if $pw(A,B)$ is a poorly chosen password, and belongs to a finite dictionary $\mathcal{D}$, then the challenge–response transaction can be attacked in other ways: the intruder guesses $d \in \mathcal{D}$ and tries to decrypt both messages 0 and 1 with d. He obtains two values, v_0 and v_1 respectively. If $v_1 = v_0 + 1$, then the attacker has guessed the correct value $d = K$ with high probability.

This dictionary attack has been used to exploit systems in the past, often quite successfully. Note that using probabilistic encryption do not allow to prevent the attack described above since the attack uses only the decryption algorithm which is necessarily deterministic.

2.3. Enhanced Kerberos Protocol

As outlined in (Gong et al., 1993), the Kerberos protocol (Steiner et al., 1988) contains some messages which make it vulnerable to known–plaintext attacks. To avoid this problem, (Gong et al., 1993) propose the following modification, which we shall use as a running example. Like in Section 2.2, probabilistic encryption do not prevent the attack described below, hence, for sake of clarity, we shall use below the deterministic encryption symbol with only two parameters.

$$\begin{array}{lll} 0.\ A \rightarrow S: & \{A, B, N_1, N_2, Ca, \{Ta\}_{pw(A,S)}\}_{pub(S)} \\ 1.\ S \rightarrow A: & \{N_1, K \oplus N_2\}_{pw(A,S)}, \{A, K, Ts\}_{pw(B,S)} \\ 2.\ A \rightarrow B: & \{A, K, Ts\}_{pw(B,S)} \end{array}$$

In this protocol, the user A obtains from S a secret key K to be shared between himself and the ticket–granting service B. Afterward, A can obtain tickets for other services from B using this key K. The symbol $\oplus$ denotes the bit–wise exclusive or operation. We do not consider any algebraic properties of this operation here and rather see it as an encryption: $K \oplus N$ is equivalent to $\{K\}_N$. $pub(S)$ is the public–key of the server S, and $pw(A, S)$, $pw(B, S)$ are symmetric keys (passwords) that A and B respectively share with S.

The password $pw(B, S)$ can be assumed to be well–chosen since B is a server, but the password $pw(A, S)$ of the user A is likely to come from a dictionary. This protocol implements some protections against dictionary attacks on $pw(A, S)$, using the nonces N_1, N_2, the confounder Ca (which is a long nonce whose role is to confound attacks like in Section 2.2), and the timestamps Ta and Ts, added in order to prevent the replay of messages 0 and 1. We refer the reader to (Gong et al., 1993) for the details about this protocol.

As described in (Tsudik and Herreweghen, 1993; Gong et al., 1993; Lowe, 2004) for similar protocols, if the server S does not record the timestamp Ta, and if moreover the clocks of S and A are not well synchronized, an intruder can replay a copy of an eavesdropped message 0 within the clock skew, making possible the attack described in Figure 1.

In the session β, the intruder replays A's message $\alpha.0$, so as to get the server to issue (in $\beta.1$) another message using the same nonces N_1 and N_2. Hence, N_1 can be used as a verifier to guess $pw(A, S)$: the intruder can decrypt $\{N_1, K \oplus N_2\}_{pw(A,S)}$ and $\{N_1, K' \oplus N_2\}_{pw(A,S)}$ with a value d chosen in a dictionary, and if the first field of the two values obtained is the same, then it means that the value guessed $d = pw(A, S)$. After

$\alpha.0.\ A \rightarrow S:\quad \{A, B, N_1, N_2, Ca, \{Ta\}_{pw(A,S)}\}_{pub(S)}$
$\alpha.1.\ S \rightarrow A:\quad \{N_1, K \oplus N_2\}_{pw(A,S)}, \{A, K, Ts\}_{pw(B,S)}$

$\beta.0.\ I(A) \rightarrow S:\ \{A, B, N_1, N_2, Ca, \{Ta\}_{pw(A,S)}\}_{pub(S)}$
$\beta.1.\ S \rightarrow I(A):\ \{N_1, K' \oplus N_2\}_{pw(A,S)}, \{A, K', Ts'\}_{pw(B,S)}$

I guesses $pw(A,S)$ offline

$\gamma.0.\ I(A) \rightarrow S:\ \{A, B, M_1, M_2, Ci, \{Ti\}_{pw(A,S)}\}_{pub(S)}$
$\gamma.1.\ S \rightarrow I(A):\ \{M_1, K'' \oplus M_2\}_{pw(A,S)}, \{A, K'', Ts''\}_{pw(B,S)}$
$\gamma.2.\ I(A) \rightarrow B:\ \{A, K'', Ts''\}_{pw(B,S)}$

The notation $I(A)$ represents: on the left hand side of an arrow, the intruder I masquerading A to send a message, and on the right hand side of an arrow, the intruder intercepting a message intended for A.

Figure 1. The description of the attack on the Kerberos protocol.

that, the intruder can impersonate A in a third session γ, with chosen nonces M_1 and M_2, and he obtains in $\gamma.1$ the session key K'' which is assumed to be a secret shared between A, S and B.

3. Abstract Model for Protocols

We assume given a signature $\mathcal{F}$ containing the symbols $\langle \_, \_ \rangle$ (pairing[1]), $\{\_\}_\_$ (deterministic encryption), $\{\_\}^-_\_$ (probabilistic encryption), some unary function symbols representing invertible functions, others representing one way functions, and three other special constructors: $pw(\_)$ (for symmetric keys, or passwords shared between agents), $pub(\_)$ (for asymmetric public keys of agents), and $priv(\_)$ (for the corresponding private keys).

The signature $\mathcal{F}$ contains also an arbitrary subset $\mathcal{F}_0$ of constant symbols representing objects like keys, agent names, nonces... We also assume given an infinite set of variables $\mathcal{X}$. The set of terms built with $\mathcal{F}$ and $\mathcal{X}$ is denoted by $\mathcal{T}(\mathcal{F}, \mathcal{X})$ and the subset of ground terms (terms without variables) $\mathcal{T}(\mathcal{F})$. We denote $vars(t)$ the set of variables occurring in a term $t \in \mathcal{T}(\mathcal{F}, \mathcal{X})$, and $st(t)$ the set of subterms of t. These two notations are extended as expected to a structure T containing some terms: $vars(T)$ (resp. $st(T)$) is the union of the sets $vars(t)$ (resp. $st(t)$) for every term t occurring in T.

[1] For sake of simplicity we usually write t_1, t_2 instead of $\langle t_1, t_2 \rangle$.

We assume a linear well–founded ordering $\prec$ on the ground terms of $\mathcal{T}(\mathcal{F})$ such that the constant 0 is minimal w.r.t. $\prec$. We shall use below the (well–founded) extension $\ll$ of $\prec$ to multisets of ground terms, see (Dershowitz, 1987). For sake of notations, given two substitutions σ_1 and σ_2, we write $\sigma_1 \ll \sigma_2$ iff $img(\sigma_1) \ll img(\sigma_2)$ ($img(\sigma_i)$ is the multiset of all $x\sigma$ such that $x \in dom(\sigma_i)$).

Among the terms of $\mathcal{T}(\mathcal{F})$, we distinguish a finite subset $\mathcal{G}$ of *guessable* symbols such that $\mathcal{G} \subseteq \mathcal{F}_0 \cup \{pw(t), pub(t), priv(t) | t \in \mathcal{T}(\mathcal{F})\}$. Theses symbols all have the distinctive feature of taking their values in finite (quite small) dictionaries known by everyone. Moreover, we assume a bijective mapping denoted $\_^{-1}$ from $\mathcal{T}(\mathcal{F},\mathcal{X})$ into $\mathcal{T}(\mathcal{F},\mathcal{X})$ which associates to a public key the corresponding private key and reciprocally. More precisely, if $k \in \mathcal{F}_0$ represents an asymmetric key, public or private, then $k^{-1} \in \mathcal{F}_0$ represents its private (resp. public) counterpart. For every $t \in \mathcal{T}(\mathcal{F},\mathcal{X})$, we have $pub(t)^{-1} = priv(t)$ and $priv(t)^{-1} = pub(t)$, and for every other $s \in \mathcal{T}(\mathcal{F},\mathcal{X})$ (which does not represent a public or private key), we have $s^{-1} = s$.

A *substitution* is the term morphism extension of a finite mapping $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ where $x_1, \ldots, x_n \in \mathcal{X}$ and $t_1, \ldots t_n \in \mathcal{T}(\mathcal{F},\mathcal{X})$. If $t_1, \ldots t_n \in \mathcal{T}(\mathcal{F})$, the substitution is called *ground*. As usual, the application of a substitution σ to a term t and the composition of substitutions σ_1 by σ_2 are written in postfix notation, respectively $t\sigma$ and $\sigma_1\sigma_2$. A substitution σ is *grounding for* t if $t\sigma \in \mathcal{T}(\mathcal{F})$. Given two terms u and v the replacement of u by v, denoted by $[u \mapsto v]$, maps every term t to the term $t[u \mapsto v]$ which is obtained by replacing all occurrences of u in t by v. Note that the result of such replacement is uniquely determined.

In the paper, $|S|$ denotes the cardinal of the set S. The *size* $\|t\|$ of a term t is the number of nodes in t. This notation is extended as expected to a set of terms $\|T\|$. The *dag–size* $\|T\|_d$ of a term container T is the number of distinct subterms of T (*i.e.* the number of nodes in a representation of T as a dag with maximal sharing). More details about the dag representations of terms can be found in (Rusinowitch and Turuani, 2001).

3.1. Protocol Syntax

Definition 1. *(protocol)*
A protocol *is a finite set of programs, each program being a finite sequence of pairs of* instructions *of the form* $\mathsf{recv}(r); \mathsf{send}(s)$ *with* $r, s \in \mathcal{T}(\mathcal{F},\mathcal{X})$*.*

It is semantically equivalent to consider programs which are sequences of send and recv in an arbitrary order, since we may add some instructions $\mathsf{send}(0)$ and $\mathsf{recv}(0)$ where $0 \in \mathcal{F}_0 \setminus \mathcal{G}$ is a special constant known to everyone.

The formalism used here to model protocols is close to so–called strand spaces (Thayer et al., 1999) and is sufficiently general to describe a lot of protocols. However, like the strand space model, it suffers from some drawbacks. For instance, it does not permit to model an agent who receives a ciphertext and who later verifies after he receives the decryption key.

Example 1. *(Kerberos protocol)*
The Kerberos protocol variant described in Section 2.3 is made of three programs:

$$
\begin{array}{lll}
\textit{role } A: & 0. & \mathsf{recv}(0);\ \mathsf{send}(\{x^0_A, x^0_B, x^0_{N_1}, x^0_{N_2}, x^0_{Ca}, \{x^0_{Ta}\}_{x^0_{pw(A,S)}}\}_{x^0_{pub(S)}}) \\
 & 1. & \mathsf{recv}(\{x^0_{N_1}, x^0_K \oplus x^0_{N_2}\}_{x^0_{pw(A,S)}}, x^0);\ \mathsf{send}(x^0) \\
\textit{role } S: & 0. & \mathsf{recv}(\{x^1_A, x^1_B, x^1_{N_1}, x^1_{N_2}, x^1_{Ca}, \{x^1_{Ta}\}_{pw(x^1_A, x^1_S)}\}_{x^1_{pub(S)}}); \\
 & & \mathsf{send}(\{x^1_{N_1}, x^1_K \oplus x^1_{N_2}\}_{pw(x^1_A, x^1_S)}, \{x^1_A, x^1_K, x^1_{Ts}\}_{pw(x^1_B, x^1_S)}) \\
\textit{role } B: & 0. & \mathsf{recv}(\{x^2_A, x^2_K, x^2_{Ts}\}_{x^2_{pw(B,S)}});\ \mathsf{send}(0)
\end{array}
$$

The symbols $x^i_A \ldots$ ($i = 0, 1, 2$) are distinct variables of $\mathcal{X}$. Note that A receives (in step 1) the ciphertext $\{A, K, Ts\}_{pw(B,S)}$ as a value x^0, and forwards it blindly (to B), since he does not know B's password $pw(B, S)$. The program of role B implements only the reception of the last message by B.

3.2. Protocol Semantics

Definition 2. *(process)*
Given a protocol $\mathcal{P}$, a process (p, σ) *following $\mathcal{P}$ is made of a program $p \in \mathcal{P}$ and a ground substitution σ whose domain is a subset of $vars(p)$.*

Every program of the protocol $\mathcal{P}$ defines a *role*, and a process (p, σ) is an honest *agent* playing the role p. A *configuration* is a pair (S, N) where S is a finite set of processes whose programs have disjoint sets of variables, and N is a set of ground terms representing the messages currently in the network (the message sent and not yet received).

We define operational semantics for the execution of processes. Each step changes the running configuration, denoted $(S, N) \rightharpoonup_i (S', N')$ for some $0 \le i < |S|$, if the i^{th} process of S is (p, σ) with $p = \mathsf{recv}(r); \mathsf{send}(s); p'$ and:

– the instruction $\mathsf{recv}(r)$ is executed properly, *i.e.* there exists a ground substitution θ such that $r\sigma\theta \in N$, S' is obtained from S

by updating the i^{th} process to (p', σ'), where $\sigma' = \sigma\theta$, and leaving the other process unchanged, and the message $r\theta$ is later removed from the network,

– the instruction $\mathsf{send}(s)$ is executed, by adding the term $s\sigma'$ to the network, *i.e.* $N' = (N \setminus \{r\theta\}) \cup \{s\sigma'\}$.

We assume that the protocol and the initial configuration are such that after every such step, $s\sigma'$ is ground. It means that the messages sent are built from values completely defined by the previous steps of the protocol and the initial knowledge of the processes. More formally:

Definition 3. *(runnable)*
An initial configuration $(\{(p_0, \sigma_0), \ldots, (p_m, \sigma_m)\}, N_0)$ *of a protocol* $\mathcal{P}$ *is called* runnable *iff for each* $i \leq m$ *such that the program* p_i *is the sequence* $(\mathsf{recv}(r_{i,j}); \mathsf{send}(s_{i,j}))_{j \leq n}$*, for each* $j \leq n$*, we have:*

$$vars(s_{i,j}) \subseteq dom(\sigma_i) \cup \bigcup_{k<j} vars(r_{i,k})$$

Example 2. *(Kerberos protocol)*
The set of processes $((p_0, \sigma_0),\ (p_1, \sigma_1),\ (p_2, \sigma_2))$ *described below is the first component of a runnable initial configuration of the protocol of Example 1:*

$$\sigma_0 = \left\{ \begin{array}{l} x^0_A \mapsto A, x^0_B \mapsto B, x^0_S \mapsto S, x^0_{Ca} \mapsto Ca, \\ x^0_{N_1} \mapsto N_1, x^0_{N_2} \mapsto N_2, x^0_{Ta} \mapsto Ta, \\ x^0_{pub(S)} \mapsto pub(S), x^0_{pw(A,S)} \mapsto pw(A,S), \end{array} \right\}$$

$$\sigma_1 = \left\{ \begin{array}{l} x^1_K \mapsto K, x^1_{Ts} \mapsto Ts, \\ x^1_S \mapsto S, x^1_{pub(S)} \mapsto pub(S) \end{array} \right\}$$

$$\sigma_2 = \{x^2_{pw(B,S)} \mapsto pw(B,S)\}$$

where A*,* B*,* S*,* N_1*,* N_2*,* Ca*,* Ta*,* K*,* Ts *are constants of* $\mathcal{F}_0$*.*

In this paper, we are interested in proving the reachability (from a given runnable initial configuration) of a configuration where some security property is compromised, assuming the presence of an intruder who has some control over the communication network, as defined in the next section.

Axiom (A)	$\dfrac{u \in T}{T \vdash u}$	Pairing (P)	$\dfrac{T \vdash u \quad T \vdash v}{T \vdash \langle u, v\rangle}$
Unpairing (UL)	$\dfrac{T \vdash \langle u, v\rangle}{T \vdash u}$	Unpairing (UR)	$\dfrac{T \vdash \langle u, v\rangle}{T \vdash v}$
Encryption (E)	$\dfrac{T \vdash u \quad T \vdash v}{T \vdash \{u\}_v}$	Decryption (D)	$\dfrac{T \vdash \{u\}_v \quad T \vdash v^{-1}}{T \vdash u}$

Figure 2. The Standard Dolev–Yao Model.

4. Intruder Model

We consider now an intruder with some control over the communication network. He is able to *read* any message in the network and to analyze the messages read in order to deduce new messages, using some inference rules like the ones defined in the following subsections 4.1, 4.2, and 4.3. When the abilities of the intruder are limited to the two former actions, we call him a *passive* intruder. The problems of the complexity of intruder deduction, as well as protocol verification in presence of a passive intruder with guessing abilities are studied in Section 6.

The intruder can also be assumed to be able to *send* new messages to the network: he is then called an *active* intruder. The verification of protocol insecurity in presence of an active intruder is studied in Section 7 and 8.

We shall give more formal definitions of passive and active intruders in Section 4.6.

4.1. Dolev–Yao Deduction Model

The most widely used deduction relation representing the analysis abilities of an intruder is often refereed as *Dolev–Yao model* after (Dolev and Yao, 1983), though the formalism of this paper is not exactly the same as the one used here. It is denoted by a "sequent" $T \vdash u$ (the intruder is able to deduce the term $u \in \mathcal{T}(\mathcal{F})$ from the finite set of terms $T \subseteq \mathcal{T}(\mathcal{F})$), and is defined by the inference rules in Figure 2.

In this model, the intruder can form pairs and ciphertexts from known terms (rules P, E), decompose pairs (rules UL, UR), and decrypt ciphertexts, providing that he can deduce the decryption key (rules D). The latter condition is known as the *perfect cryptography assumption*.

$$\mathsf{Enc.\ (Ep)}\ \frac{T \vdash u \quad T \vdash v \quad T \vdash r}{T \vdash \{u\}_v^r} \qquad \mathsf{Dec.\ (Dp)}\ \frac{T \vdash \{u\}_v^r \quad T \vdash v^{-1}}{T \vdash u}$$

Figure 3. Probabilistic Encryption *à la* Dolev–Yao.

4.2. Probabilistic Encryption

We have seen the importance of the probabilistic encryption primitive in the context of dictionary attacks (see Section 2.1). We propose here an extension of the above Dolev–Yao model by adding two inference rules, described in Figure 3, to deal with probabilistic encryption. The rule (Ep) says that the intruder can form a ciphertext from a known term (plaintext) u, and a random value r. The ciphertext depends on the input r. The rule (Dp) do not depends on the random input r: a decryption algorithm is necessarily deterministic.

The Dolev–Yao model of Section 4.1 extended with the two inference rules displayed in Figure 3 is called the *Dolev–Yao model with probabilistic encryption*.

4.3. Guessing Model

We shall now describe how dictionary attacks can be modeled as an extension of the above Dolev–Yao model. For this purpose, we refer to the following definition of dictionary attacks from (Lowe, 2004) that claims to generalize the definition of (Gong et al., 1993):

> A dictionary attack consists of the intruder guessing a value d, and then verifying it. The verification will be by the intruder using d to produce a value v, which we call the verifier and can take a number of different forms:
>
> 1. the intruder knew v initially, (see Section 2.1)
> 2. the intruder produced v in two distinct ways from d, (see Section 2.2)
> 3. v is an asymmetric key, and the intruder knows v's inverse from somewhere.

Intuitively, the intruder knows that $g \in \mathcal{G}$ belongs to a dictionary, in which he picks d. If the verifier v, built with d and the intruder's knowledge ensures one of the three conditions above, then the probability is high that $d = g$. We shall use a variant of the rules of Figure 2 in order to model the guessing of a d, and the production of a verifier $v \in \mathcal{T}(\mathcal{F})$ by the intruder. In the rules of this variant, presented in Figure 4, we

$$\text{Axiom} \quad \frac{u \in T}{T/\emptyset \vdash' u}\,(\mathsf{A}') \qquad \text{Guess} \quad \frac{u \in \mathcal{G}}{T/u \vdash' u}\,(\mathsf{G})$$

$$\text{Pairing} \quad \frac{T/T_1' \vdash' u \quad T/T_2' \vdash' v}{T/T_1' \cup T_2' \vdash' \langle u, v\rangle}\,(\mathsf{P}')$$

$$\text{Encryption} \quad \frac{T/T_1' \vdash' u \quad T/T_2' \vdash' v}{T/T_1' \cup T_2' \vdash' \{u\}_v}\,(\mathsf{E}')$$

$$\text{Unpairing} \quad \frac{T/T' \vdash' \langle u, v\rangle}{T/T' \vdash' u}\,(\mathsf{UL}') \quad \frac{T/T' \vdash' \langle u, v\rangle}{T/T' \vdash' v}\,(\mathsf{UR}')$$

$$\text{Decryption} \quad \frac{T/T_1' \vdash' \{u\}_v \quad T/T_2' \vdash' v^{-1}}{T/T_1' \cup T_2' \vdash' u}\,(\mathsf{D}')$$

$$\text{Enc.} \quad \frac{T/T_1' \vdash' u \quad T/T_2' \vdash' v \quad T/T_3' \vdash' r}{T/T_1' \cup T_2' \cup T_3' \vdash' \{u\}_v^r}\,(\mathsf{Ep}')$$

$$\text{Dec.} \quad \frac{T/T_1' \vdash' \{u\}_v^r \quad T/T_2' \vdash' v^{-1}}{T/T_1' \cup T_2' \vdash' u}\,(\mathsf{Dp}')$$

Figure 4. The Dolev–Yao' model with probabilistic encryption.

introduce a new form of sequent $T/T' \vdash' v$, which means that if the intruder knows the messages in $T \subseteq \mathcal{T}(\mathcal{F})$ and guessed values for the symbols of $T' \subseteq \mathcal{G}$, then he can build the verifier $v \in \mathcal{T}(\mathcal{F})$. In other words, he can deduce that v belongs to a finite set that he can compute. The members of T and T' are respectively called the *strong* and *weak hypotheses* of $T/T' \vdash' v$, and v is called its *target*.

Figure 5 introduces a deduction rule Compare which models the verification of a guess d with one of the 3 cases described above. The conditions (i) and (ii) ensure that one of the proofs P_1 or P_2 really uses the guessable value g and it is necessary to prevent certain false attacks (see Section 4.5). The normality condition (ii) will prohibit deduction steps that simply undo previous steps. It refers to the notion of *normal* DY'–proof which is formally defined in Section 4.4. Lastly, (iii) ensures that the two proofs P_1 and P_2 do not end with the same instance of the same rule. Hence, the two ways to obtain the verifier v are really distinct.

$$\cfrac{P_1\left\{\ \cfrac{\cfrac{...}{\vdash' u_1}\quad ...\quad \cfrac{...}{\vdash' u_{n_1}}}{T/T_1'\vdash' u}\,(R_1)\right.\qquad P_2\left\{\ \cfrac{\cfrac{...}{\vdash' v_1}\quad ...\quad \cfrac{...}{\vdash' v_{n_2}}}{T/T_2'\vdash' v}\,(R_2)\right.}{T\vdash g}\,(\mathsf{C})$$

where:
(i) $g \in T_1' \cup T_2'$
(ii) P_1 and P_2 are normal DY'–proofs
(iii) $\mathsf{R}_1 \neq \mathsf{R}_2$ or $\{u, u_1, .., u_{n_1}\} \neq \{v, v_1, ..., v_{n_2}\}$
(iv) $u = v$ or $u = v^{-1}$

Figure 5. The Compare Rule.

The inference system made of the rules displayed in Figures 2, 3 and 4 and the rule Compare of Figure 5 is called the *guessing model*.

4.4. Proof Trees

Definition 4. *(proof)*
A DY–proof *(resp.* DY'–proof*) P of $T \vdash u$ (resp. $T/T' \vdash' u$) is a tree such that:*

- *every leaf of P is labeled with some $v \in T$ (resp. $v \in T \cup \mathcal{G}$),*

- *for every node n labeled with s with k sons labeled with $s_1, \ldots, s_k$, $(s_1, \ldots, s_k, s)$ is an instance of an inference rule R of Figure 2, 3 (resp. Figure 4) which $s_1, \ldots, s_k$ are the premises and s the conclusion. We say that P contains the instance $(s_1, \ldots, s_k, s)$, or ends with this instance (or simply with rule R) if n is the root of P,*

- *the root is labeled with some $T \vdash u$ (resp. $T/T' \vdash' u$).*

A guessing–proof *is a tree ending with an instance of the rule* Compare *and whose two sons are DY'–proofs which satisfy the conditions (i)–(iv) of Figure 5.*

Let P be a DY–proof (resp. DY'–proof). We say that the pair $\langle u_1, u_2\rangle$ is *decomposed* in P if P contains an instance of the rule (UL) or (UR) (resp. (UL') or (UR')) whose target of the premise is $\langle u_1, u_2\rangle$. Similarly, the ciphertext $\{u_1\}_{u_2}$ (resp. $\{u_1\}_{u_2}^{u_3}$) is said decomposed in P if P contains an instance of the rule (D) or (D') (resp. (Dp) or (Dp')) whose targets of premises are $\{u_1\}_{u_2}$ and u_2^{-1} (resp. $\{u_1\}_{u_2}^{u_3}$, u_2^{-1} and u_3).

Definition 5. *(minimality)*
A DY–proof or DY'–proof P is called minimal *if it does not contain two nodes on the same path labeled by sequents with the same target.*

The following notion of normal proof is used in the definition of the rule $\mathsf{Compare}$.

Definition 6. *(normality)*
A DY'–proof P is called normal *if the rewrite rules defined in Figure 6 can not be applied to P.*

The normality of DY–proofs is defined with a similar set of rewrite rules, where (D', E', P', UL', UR', Dp', Ep') are replaced respectively by (D, E, P, UL, UR, Dp, Ep).

The rewrite rules of Figure 6 correspond to algebraic properties of the operators. For instance, the first rule expresses that the composition of encryption and decryption (with the keys of the same keypair) is the identity. The third rewrite rule states that this is also the case of the composition of decryption and deterministic encryption. To be more precise, the rewrite rules described in Figure 6 are used to model the algebraic properties listed below. The pairing, the left and right projection operators are respectively denoted P, UL, UR. The encryption and decryption operators, and their probabilistic counterparts, are respectively denoted E, D, Ep, Dp.

- $\mathsf{D}(\mathsf{E}(x, y), y^{-1}) = x$ and $\mathsf{E}(\mathsf{D}(x, y^{-1}), y) = x$,
- $\mathsf{Dp}(\mathsf{Ep}(x, y, z), y^{-1}) = x$,
- $\mathsf{UL}(\mathsf{P}(x_1, x_2)) = x_1$, $\mathsf{UR}(\mathsf{P}(x_1, x_2)) = x_2$ and $\mathsf{P}(\mathsf{UL}(x), \mathsf{UR}(x)) = x$.

Note that there is no analogous of the rule $\mathsf{E}(\mathsf{D}(x, y^{-1}), y) = x$, for Ep and Dp. Indeed, while the decryption of a ciphertext $\{u_1\}^r_{u_2}$ returns u_1, the probabilistic re–encryption of u_1 returns $\{u_1\}^{r'}_{u_2}$, with $r' \neq r$ and hence this ciphertext differs from the original one. In the setting of dictionary attacks, it is crucial to be able to find proof which are not equivalent modulo these rewrite rules, as explained in Section 4.5.

Note that every minimal DY'–proof is normal. Indeed, every left member of rewrite rule of Figure 6 contains two nodes on the same path labeled by sequents with the same target (but not necessarily with the same set of weak hypotheses).

Lemma 1. *If there exists a DY–proof of $T \vdash u$, then there exists a minimal DY–proof of $T \vdash u$.*

$$\cfrac{\cfrac{\cfrac{Q_1}{T/T_1' \vdash' u_1} \quad \cfrac{Q_2}{T/T_2' \vdash' u_2}}{T/T_1' \cup T_2' \vdash' \{u_1\}_{u_2}}\,(\mathsf{E}') \qquad \cfrac{Q_3}{T/T_3' \vdash' u_2^{-1}}}{T/T_1' \cup T_2' \cup T_3' \vdash' u_1}\,(\mathsf{D}') \quad \rightarrow \quad \cfrac{Q_1}{T/T_1' \vdash' u_1}$$

$$\cfrac{\cfrac{\cfrac{Q_1}{T/T_1' \vdash' u_1} \quad \cfrac{Q_2}{T/T_2' \vdash' u_2} \quad \cfrac{Q_3}{T/T_3' \vdash' r}}{T/T_1' \cup T_2' \cup T_3' \vdash' \{u_1\}^r_{u_2}}\,(\mathsf{Ep}') \qquad \cfrac{Q_4}{T/T_4' \vdash' u_2^{-1}}}{T/T_1' \cup T_2' \cup T_3' \cup T_4' \vdash' u_1}\,(\mathsf{Dp}') \quad \rightarrow \quad \cfrac{Q_1}{T/T_1' \vdash' u_1}$$

We also have an additional rule for the deterministic encryption:

$$\cfrac{\cfrac{\cfrac{Q_1}{T/T_1' \vdash' \{u_1\}_{u_2}} \quad \cfrac{Q_2}{T/T_2' \vdash' u_2^{-1}}}{T/T_1' \cup T_2' \vdash' u_1}\,(\mathsf{D}') \qquad \cfrac{Q_3}{T/T_3' \vdash' u_2}}{T/T_1' \cup T_2' \cup T_3' \vdash' \{u_1\}_{u_2}}\,(\mathsf{E}') \quad \rightarrow \quad \cfrac{Q_1}{T/T_1' \vdash' \{u_1\}_{u_2}}$$

Concerning the pairing symbol, we have the following rules:

$$\cfrac{\cfrac{\cfrac{Q_1}{T/T_1' \vdash' u_1} \quad \cfrac{Q_2}{T/T_2' \vdash' u_2}}{T/T_1' \cup T_2' \vdash' \langle u_1, u_2 \rangle}\,(\mathsf{P}')}{T/T_1' \cup T_2' \vdash' u_1}\,(\mathsf{UL}') \quad \rightarrow \quad \cfrac{Q_1}{T/T_1' \vdash' u_1}$$

and a similar rewrite rule for (P',UR').

$$\cfrac{\cfrac{\cfrac{Q_1}{T/T_1' \vdash' \langle u_1, u_2 \rangle}}{T/T_1' \vdash' u_1}\,(\mathsf{UL}') \qquad \cfrac{\cfrac{Q_2}{T/T_2' \vdash' \langle u_1, u_2 \rangle}}{T/T_2' \vdash' u_2}\,(\mathsf{UR}')}{T/T_1' \cup T_2' \vdash' \langle u_1, u_2 \rangle}\,(\mathsf{P}') \quad \rightarrow \quad \cfrac{Q_i}{T/T_i' \vdash' \langle u_1, u_2 \rangle}$$

for $i = 1, 2$.

Figure 6. The proof rewrite rules.

Proof. We can show this result by induction on the proof P of $T \vdash u$. If P is reduced to an instance of the rule (A), then it is obvious. Otherwise, by induction hypothesis the direct subproofs of P are minimal. So, if P is not minimal, there exists two nodes on the same path labeled by the sequent $T \vdash u$ and one of these is the root of P. Hence, the other is the root of a minimal proof of $T \vdash u$. $\square$

However this result is not valid for DY'–proofs, unless the set of weak hypotheses is empty for every node. The problem with DY'–proofs is that we can not assume that two nodes on the same path and with the same target have the same set of weak hypotheses.

4.5. Rule Compare

In this section, we shall explain in more details the conditions of application of the rule Compare. We also provide some examples to illustrate this rule.

Condition (ii) By the condition (i), one son P_1 or P_2 contains the guess g among the weak hypotheses, but it is not sufficient to ensure that P_1 or P_2 really depends on the guess g: only the condition (ii) ensures this property. Indeed, without this condition, there would be a guessing–proof with the two (non normal) sons below, which would mean that the intruder is able to guess any $g \in \mathcal{G}$ from any message m known by the intruder.

$$\dfrac{m \in T}{T/\emptyset \vdash' m}\,(\mathsf{A}') \qquad \dfrac{\dfrac{\dfrac{m \in T}{T/\emptyset \vdash' m}\,(\mathsf{A}') \quad \dfrac{g \in \mathcal{G}}{T/\{g\} \vdash' g}\,(\mathsf{G})}{T/\{g\} \vdash' \langle m, g\rangle}\,(\mathsf{P}')}{T/\{g\} \vdash' m}\,(\mathsf{UL}')$$

The idea behind condition (ii) is that in a proof like the one above, re–obtaining the value m after the successive applications of pairing (P') and projection (UL') is not a commitment that the guess of g is correct. It is just an algebraic property of the operators, characterized by a rewrite rule of Figure 6.

Condition (iii) The condition (iii) also prevents certain false attacks. Consider the program which is made up of one pair of instructions $\mathsf{recv}(x); \mathsf{send}(\langle x, x\rangle)$. An agent executing this program will answer to any message m with the pair $\langle m, m\rangle$. Thus, if the intruder knows the message $m = \{n\}^r_g$, he obtains from the agent the answer $\langle \{n\}^r_g, \{n\}^r_g\rangle$. The two DY'–proofs below verify the conditions (i), (ii) and (iv) of the rule Compare. However, the condition (iii) is not verified and these two proofs indeed represent a fake dictionary attack. Intuitively, in this attack, the intruder guesses a value d for g, computes the left and right projections ((UL′) and (UR′)) of m, tries to decrypt each projection with d (rule (Dp′)) and compares the values obtained. But, since both projections are $\{n\}_g^r$, the values will always be equal, even when $d \neq g$. We see that the two proofs differ (the first step is left or right projection) but that their last instance is the same (decryption applied to the same premises).

$$\dfrac{\dfrac{\dfrac{\langle \{n\}_g^r, \{n\}_g^r \rangle \in T}{T/\emptyset \vdash' \langle \{n\}_g^r, \{n\}_g^r \rangle}\,(\mathsf{A}')}{T/\emptyset \vdash' \{n\}_g^r}\,(\mathsf{UL}') \qquad \dfrac{g \in \mathcal{G}}{T/\{g\} \vdash' g}\,(\mathsf{G})}{T/\{g\} \vdash' n}\,(\mathsf{Dp}')$$

$$\dfrac{\dfrac{\dfrac{\langle \{n\}_g^r, \{n\}_g^r \rangle \in T}{T/\emptyset \vdash' \langle \{n\}_g^r, \{n\}_g^r \rangle}\,(\mathsf{A}')}{T/\emptyset \vdash' \{n\}_g^r}\,(\mathsf{UR}') \qquad \dfrac{g \in \mathcal{G}}{T/\{g\} \vdash' g}\,(\mathsf{G})}{T/\{g\} \vdash' n}\,(\mathsf{Dp}')$$

Note that if the two ciphertexts of the pair would have been obtained by two distinct applications of the probabilistic encryption algorithm, we would have obtained a pair which is composed of two distinct terms, *i.e.* $\langle \{n\}_g^r, \{n\}_g^{r'} \rangle$, allowing surprisingly the intruder to mount a dictionary attack.

We illustrate the application of the rule Compare on the protocols given in Section 2.

Example 3. *(naive vote protocol)*
Let $T = \{\{V\}_{pub(S)}, pub(S)\}$ *and* $\mathcal{G} = \{V\}$*. The following guessing–proof models the guessing attack performed by the intruder on* V*:*

$$\dfrac{\dfrac{\dfrac{V \in \mathcal{G}}{T/\{V\} \vdash V}\,(\mathsf{G}) \qquad \dfrac{pub(S) \in T}{T/\emptyset \vdash' pub(S)}\,(\mathsf{A}')}{T/\{V\} \vdash' \{V\}_{pub(S)}}\,(\mathsf{E}') \qquad \dfrac{\{V\}_{pub(S)} \in T}{T/\emptyset \vdash' \{V\}_{pub(S)}}\,(\mathsf{A}')}{T \vdash V}\,(\mathsf{C})$$

Example 4. *(handshake protocol)*
Let $T = \{\{N\}_{pw(A,B)}^{ra}, \{N+1\}_{pw(A,B)}^{rb}\}$ *and* $\mathcal{G} = \{pw(A,B)\}$*. The required guessing–proof of* $T \vdash pw(A,B)$ *can be obtained by applying*

the rule Compare *to the two DY'–proofs below:*

$$\dfrac{\dfrac{\dfrac{\{N\}_{pw(A,B)} \in T}{T/\emptyset \vdash' \{N\}^{ra}_{pw(A,B)}}\,(\mathsf{A}') \qquad \dfrac{pw(A,B) \in \mathcal{G}}{T/\{pw(A,B)\} \vdash' pw(A,B)}\,(\mathsf{G})}{T/\{pw(A,B)\} \vdash' N}\,(\mathsf{Dp}')}{T/\{pw(A,B)\} \vdash' N+1}$$

$$\dfrac{\dfrac{\{N+1\}^{rb}_{pw(A,B)} \in T}{T/\emptyset \vdash' \{N+1\}_{pw(A,B)}}\,(\mathsf{A}') \qquad \dfrac{pw(A,B) \in \mathcal{G}}{T/\{pw(A,B)\} \vdash' pw(A,B)}\,(\mathsf{G})}{T/\{pw(A,B)\} \vdash' N+1}\,(\mathsf{Dp}')$$

Note that the conditions (i)–(iv) required to apply the rule Compare *are satisfied.*

4.6. Passive and Active Intruder

A ground term $u \in \mathcal{T}(\mathcal{F})$ is called *deducible* from a set of ground terms $N \subseteq \mathcal{T}(\mathcal{F})$ if and only if there exists a finite subset T of N and a DY–proof of $T \vdash u$. We note $ded(N)$ the set of ground terms deducible from N.

In order to define the transition performed by the intruder, we extend the configurations of the system with a third component representing the intruder's knowledge. Therefore, a configuration is a triple (S, N, K) where S and N are as in Section 3.2 and $K \subseteq \mathcal{T}(\mathcal{F})$. The transitions of the intruder, denoted $\rightharpoondown$, are defined as follows:

listen	$(S, N, K) \rightharpoondown (S, N, K \cup \{s\})$ for some $s \in N$
deduce	$(S, N, K) \rightharpoondown (S, N, K \cup \{s\})$ if $s \in ded(K)$
write	$(S, N, K) \rightharpoondown (S, N \cup \{s\}, K)$ for some $s \in K$
divert	$(S, N, K) \rightharpoondown (S, N \setminus \{s\}, K)$ for some $s \in N$

We define a *passive intruder* as able of performing the transitions listen and deduce and an *active intruder* as able of performing the two latter transitions and moreover write and divert.

Finally, we define the relation $\rightharpoondown_g$, for $g \in \mathcal{G}$, by $(S, N, K) \rightharpoondown_g (S, N, K \cup \{g\})$, if and only if there exists a finite subset T of K and a guessing–proof of $T \vdash g$.

5. Security Problems

The first problem, we are interested in, is the so–called intruder deduction problem, which is the following one:

Definition 7. *(Intruder Deduction problem (ID))*

IN: a finite set of terms T, a finite set of guessable values $\mathcal{G}$, a term s (the secret).

OUT: Is s deducible from T knowing that $\mathcal{G}$ are guessable values? In other words, does there exists a proof of $T \vdash s$ in the guessing model ?

There are two factors of non determinism in the semantics of protocols and intruder defined respectively in Sections 3 and 4:

- the selection of the process who is going to proceed,
- the choice, by the intruder, of a value that he will be trying to guess offline, and when he makes the offline dictionary attack.

The following definition of interleavings is a characterization of these two choices.

Definition 8. *(interleaving)*
Given an initial configuration $\mathcal{S} = (\{(p_0, \sigma_0), \ldots, (p_m, \sigma_m)\}, N_0)$ of a protocol $\mathcal{P}$, an interleaving *of $\mathcal{S}$ is a finite sequence I of values which can be either terms of $\mathcal{G}$ or integers in $0...m$ (indices of processes of $\mathcal{S}$) such that the number of occurrences of every index i in I is smaller than or equal to the number of instruction pairs in p_i.*

Hence, an interleaving defines a scenario for processes execution and intruder guessing in a potential attack. The problem of trace insecurity consists in deciding whether such a scenario can be directed by the intruder, and whether, when following this scenario, the system reaches a critical state.

We define the relation $\rightarrow_i$ on configurations, for an integer i, by $\rightarrow_i := \rightharpoondown^* \rightharpoonup_i \rightharpoondown^*$ ($\rightharpoondown^*$ denotes the reflexive and transitive closure of $\rightharpoondown$, and $\rightharpoonup_i$ is extended to triples as expected) and we define the relation $\rightarrow_g$, for $g \in \mathcal{G}$, by $\rightarrow_g := \rightharpoondown^* \rightharpoondown_g \rightharpoondown^*$.

The relation $\rightarrow_I$, for an interleaving I, is defined recursively as follows: $\rightarrow_\emptyset$ is the identity, $\rightarrow_{I,i} := \rightarrow_I \rightarrow_i$ for an integer i (I, i denotes the interleaving obatined by appending i to I) and $\rightarrow_{I,g} := \rightarrow_I \rightarrow_g$ for $g \in \mathcal{G}$.

Definition 9. *(Trace Insecurity problem (TI))*

IN: a protocol $\mathcal{P}$, a runnable initial configuration (S_0, N_0) of $\mathcal{P}$, a finite set $K_0 \subset \mathcal{T}(\mathcal{F})$ (the initial knowledge of the intruder), an interleaving I of (S_0, N_0), and a ground term $s \in \mathcal{T}(\mathcal{F})$.

OUT: does there exists (S, N, K) *such that* $(S_0, N_0, K_0) \rightarrow_I (S, N, K)$ *and* s *is deducible from* K*?*

We study in Section 6, the case of a passive intruder. In particular, we prove that the intruder deduction problem is decidable in PTIME. This yields to an NP decision procedure for the trace insecurity problem in presence of a passive intruder. Sections 7 and 8 are devoted to prove that the trace insecurity problem is NP–complete when the intruder is active.

Remark 1. *For sake of simplicity, the signature* $\mathcal{F}$ *and the set* $\mathcal{G}$ *of* guessable *terms are assumed fixed, hence their size is a constant in the complexity results below. However, the complexity results would remain the same if* $\mathcal{G}$ *would be part of the input of the problem.*

We can express several trace properties of protocols as instances of (TI), like for instance failure of authentication (one process p completes the protocol presumably with an interlocutor p' whereas p' did not even start to run, and hence p has been fooled in communicating only with the intruder), or confidentiality.

The problem of the existence of an interleaving I, given $\mathcal{P}$, $\mathcal{G}$, (S_0, N_0), K_0 and s as above, returning a positive answer to (TI) is sometimes called *Protocol Insecurity* (PI) in the literature. The number of processes and their size is bounded by $\|S_0\|$, and hence the length of every possible interleaving is polynomially bounded in this measure. Therefore, a consequence of the results of Sections 6, 7 and 8 is that (PI) is decidable in NP in presence of a passive or active intruder.

6. Decision Procedures in Presence of a Passive Intruder

When the intruder is passive, the only messages circulating over the network are sent by the protocol participants (the processes). Given a protocol $\mathcal{P}$, an initial configuration (S_0, N_0), and an interleaving I, the set T of messages known by the intruder at the end is bounded and can be computed in advance (in non–deterministic polynomial time). Therefore, in this case, the trace insecurity problem (w.r.t. $\mathcal{P}$, (S_0, N_0), I, and some $K_0 \subset \mathcal{T}(\mathcal{F})$ and $s \in \mathcal{T}(\mathcal{F})$) is reducible to the existence of one DY–proof and some guessing–proofs. The complexity of the decision of the existence of (DY, DY', guessing) proofs are given in Section 6.2. Then, we give complexity results for the problems (ID, TI, PI) presented in Section 5 for the case of a passive intruder. These latter results are based on technical lemmas proved in Section 6.1.

6.1. Locality

The notion of *locality* was coined by McAllester (McAllester, 1993) to characterize theories with a deduction problem decidable in polynomial time. We follow a similar approach and prove below three locality results for, respectively, the Dolev–Yao and Dolev–Yao' model with probabilistic encryption, and the guessing model.

The proposition 1 is a locality result for DY–proofs.

Proposition 1. *A normal DY–proof P of $T \vdash u$ contains only terms in $st(T \cup \{u\})$. If moreover P ends with a decomposition rule (A, UL, UR, D, Dp), then P contains only terms in $st(T)$.*

Proof. We prove the following results simultaneously by induction on the proof P of $T \vdash u$:

1. P contains only terms in $st(T \cup \{u\})$,
2. if the last inference rule of P is a decomposition rule (A, UL, UR, D, Dp), then P contains only terms in $st(T)$.

We consider all possible cases for the last inference rule:

– rule (A), the result is straightforward.

– rule (E), then $u = \{u_1\}_{u_2}$. By induction hypothesis (1), the proof P_i ($i = 1, 2$) of $T \vdash u_i$ contains only terms in $st(T \cup \{u_i\})$. So the proof which consists of applying the rule (E) on the proofs P_1 and P_2 contains only terms in $st(T \cup \{u\})$.

– rule (P) and (Ep), it is similar to (E).

– rule (UL), then we have a normal DY–proof P_1 of $T \vdash \langle u, v\rangle$ for which the last inference rule is necessarily a decomposition rule by normality of the proof P. By induction hypothesis (2), P_1 contains only terms in $st(T)$. Hence $\langle u, v\rangle \in st(T)$, and we deduce that P contains only terms in $st(T)$.

– rule (UR), it is similar to (UL).

– rule (Dp), then we have the following derivation:

$$\dfrac{P_1\left\{\ \dfrac{\cdots}{T \vdash \{u\}_k^r}\right. \qquad P_2\left\{\ \dfrac{\cdots}{T \vdash k^{-1}}\right.}{T \vdash u}\ (\mathsf{Dp})$$

We consider the last inference rule of P_1. This rule is necessarily (A, UL, UR, D, Dp). (Ep) is impossible by normality of P. Now, we apply the induction hypothesis (2) and obtain that P_1 involves only terms in $st(T)$. We distinguish two possible cases. Firstly, if k is a symmetric key then $k^{-1} = k$. By induction hypothesis, P_2 involves only terms in $st(T \cup \{k\})$ and we have $k \in st(T)$. So, P involves only terms in $st(T)$. Secondly, if k is an asymmetric key then $k^{-1} \in \mathcal{F}_0 \cup \{pub(t), priv(t) | t \in \mathcal{T}(\mathcal{F})\}$. Consider the last inference rule of P_2. This rule is necessarily (A, UL, UR, D, Dp). By induction hypothesis, P_2 involves only terms in $st(T)$. So P involves only terms in $st(T)$.

– rule (D), it is similar to (Dp). □

The proof of Proposition 1 can be straightforwardly extended to the following analogous locality result for the DY'–proofs.

Proposition 2. *A normal DY'–proof P of $T/T' \vdash' u$ contains only terms in $st(T \cup T' \cup \{u\})$, and if P ends with a decomposition rule (G, A', UL', UR', D', Dp') then P contains only terms in $st(T \cup T')$.*

Proposition 3. *A guessing–proof of $T \vdash g$ contains only terms in $st(T \cup \mathcal{G})$.*

Proof. A guessing–proof is made up of two normal DY'–proofs P_1 of $T/T'_1 \vdash' u_1$ and P_2 of $T/T'_2 \vdash' u_2$. We distinguish two cases:

– $u_1 = u_2$
If the last inference rule of P_1 is (G, A', UL', UR', D', Dp'), then by Proposition 2, P_1 involves only terms in $st(T \cup T'_1)$, and we have $u_1 \in st(T \cup T'_1)$. So, since $T'_1 \subseteq \mathcal{G}$, we deduce that P involves only terms in $st(T \cup \mathcal{G})$.

If the last inference rule of P_2 is (G, A', UL', UR', D', Dp'), then we have also that P involves only terms in $st(T \cup \mathcal{G})$.

Otherwise both of P_1 and P_2 end with (E'), (Ep') or (P'); this case is inconsistent with the condition (iii).

– u_1 is an asymmetric key and u_2 its inverse ($u_2 = u_1^{-1}$)
In this case, $u_1, u_2 \in \mathcal{F}_0 \cup \{pub(t), priv(t) | t \in \mathcal{T}(\mathcal{F})\}$, and the last inference rule of P_1 (and P_2) is (G, A', UL', UR', D', Dp'). By Proposition 2, P_1 (resp. P_2) involves only terms in $st(T \cup T'_1)$ (resp. $st(T \cup T'_2)$), and we deduce that P involves only terms in $st(T \cup \mathcal{G})$. □

This result allows us to consider only subterms of the attacker's knowledge as potential verifiers to do a dictionary attack.

6.2. Decidability and Complexity

We show now (see Theorem 1 and Corollary 1) that when the intruder is passive, the intruder deduction problem and the trace insecurity problem (w.r.t. $\mathcal{P}$, (S_0, N_0), I, $K_0 \subset \mathcal{T}(\mathcal{F})$ and $s \in \mathcal{T}(\mathcal{F})$) can be decided by reduction to the existence of one DY–proof, for the deducibility of s (solved in Proposition 4), and several guessing–proofs(solved in Proposition 5).

Proposition 4. *Given a set of messages $T \subseteq \mathcal{T}(\mathcal{F})$, and a message $u \in \mathcal{T}(\mathcal{F})$, the existence of a DY–proof of $T \vdash u$ can be decided in polynomial time in $\|T \cup \{u\}\|_d$.*

Proof. This result follows from Lemma 1, which guarantees the existence of a minimal (hence normal) DY–proof P of $T \vdash u$, and Proposition 1, which says that P only involves terms in $st(T \cup \{u\})$. Indeed, in order to decide the existence of such a DY–proof, we construct (following (McAllester, 1993)), the set $\mathcal{S}$ of ground Horn clauses described in Figure 7 which implements a marking of every ground subterms $t \in st(T \cup \{u\})$ such that there exists a proof of $T \vdash t$. Therefore, the existence of a proof of $T \vdash u$ is equivalent to the non–satisfiability of $\mathcal{S}$, hence to the non–satisfiability of a set of proposition Horn clauses (HORN–SAT) which size is polynomial in $\|T \cup \{u\}\|_d$. Hence, the existence of a DY–proof of $T \vdash u$ can be decided in linear time in the size of $\mathcal{S}$, *i.e.* in polynomial time in $\|T \cup \{u\}\|_d$. $\square$

Proposition 5. *Given a set of messages $T \subseteq \mathcal{T}(\mathcal{F})$, and a guessable symbol $g \in \mathcal{G}$, the existence of a guessing–proof of $T \vdash g$ can be decided in polynomial time in $\|T \cup \mathcal{G}\|_d$.*

Proof. Like in the above proof of Proposition 4, we reduce the problem with the construction of a set of ground Horn clauses $\mathcal{S}'$. In order to code the instances of the rule Compare with its conditions of application, we need though to add some additional information into the atoms of the clauses.

Given a set of messages $T \subseteq \mathcal{T}(\mathcal{F})$, and a guessable symbol $g \in \mathcal{G}$, we describe in Figure 8 a set $\mathcal{S}'$ of ground Horn clauses from which one can derive the empty clause if and only if there exists a (normal) guessing–proof of $T \vdash g$. Since the size of $\mathcal{S}'$ is polynomial is $\|T \cup \mathcal{G}\|_d$, this

$$
\begin{array}{rcl|l}
 & \Rightarrow & I(u_1) & u_1 \in T \\
\\
I(u_1),\;\; I(u_2) & \Rightarrow & I(\langle u_1, u_2\rangle) & \\
I(\langle u_1, u_2\rangle) & \Rightarrow & I(u_1) & \langle u_1, u_2\rangle \in st(T \cup \{u\}) \\
I(\langle u_1, u_2\rangle) & \Rightarrow & I(u_2) & \\
\\
I(u_1),\;\; I(u_2) & \Rightarrow & I(\{u_1\}_{u_2}) & \{u_1\}_{u_2} \in st(T \cup \{u\}) \\
I(\{u_1\}_{u_2}),\;\; I({u_2}^{-1}) & \Rightarrow & I(u_1) & \\
\\
I(u_1),\;\; I(u_2),\;\; I(u_3) & \Rightarrow & I(\{u_1\}_{u_2}^{u_3}) & \{u_1\}_{u_2}^{u_3} \in st(T \cup \{u\}) \\
I(\{u_1\}_{u_2}^{u_3}),\;\; I({u_2}^{-1}) & \Rightarrow & I(u_1) & \\
\\
I(u) & \Rightarrow & &
\end{array}
$$

Figure 7. The set $\mathcal{S}$ of ground Horn clauses (proof of Proposition 4)

provides a decision procedure with the complexity wanted. The clauses of $\mathcal{S}'$ are built with two binary predicates I_0 and I_1. The ground terms occurring in the first arguments of atoms are elements of $st(T, \mathcal{G})$, and in the second argument, one of the nullary symbols A, P, E, Ep, G corresponding to rules of the system of Figure 4, or $UL(u)$, $UR(u)$, $D(u)$, $Dp(u, v)$ with $u, v \in st(T, \mathcal{G})$. The meaning of $I_0(u, l)$ is that there exists a normal DY'–proof which root is labeled with $T/T' \vdash' u$, with $T' \subseteq \mathcal{G}$, and moreover, if l is A, P, E, Ep or G then the proof ends by the corresponding inference rule, and if $l = UL(v)$, then the proof has the following form (and similarly if $l = UR(v)$ or if $l = Dp(k, r)$):

$$
\cfrac{\cfrac{\cdots}{T/T' \vdash' \langle u, v\rangle}}{T/T' \vdash' u}\;(\mathsf{UL}')
$$

This second argument l is used to ensure the conditions (ii) and (iii) of the rule Compare. The meaning of $I_1(u, l)$ is the same as $I_0(u, l)$ except that moreover $g \in T'$. □

The measure $\|\mathcal{G}\|_d$ in Proposition 5 should not be confused with the size of the dictionary. Indeed, recall that $\mathcal{G}$ is the set of guessable symbols, each of which is assume to belong to a finite dictionary. Therefore, $|\mathcal{G}|$ is rather the number of (finite) dictionaries.

Theorem 1. *The intruder deduction problem is decidable in polynomial time.*

$$
\mathcal{S}' := \left\{ \begin{array}{rcll}
 & \Rightarrow & I_1(g, G) & \\
 & \Rightarrow & I_0(u, G) & \mid\ u \in \mathcal{G} \setminus \{g\} \\
 & \Rightarrow & I_0(u, A) & \mid\ u \in T \\
I_\varepsilon(u_1, l),\ \ I_{\varepsilon'}(u_2, l') & \Rightarrow & I_{\varepsilon+\varepsilon'}(\langle u_1, u_2\rangle, P) & \mid\ l \neq UL(u_2),\ \ l' \neq UR(u_1), \\
 & & & \quad\ \langle u_1, u_2\rangle \in st(T) \\
I_\varepsilon(\langle u_1, u_2\rangle, l) & \Rightarrow & I_\varepsilon(u_1, UL(u_2)) & \mid\ l \neq P,\ \ \langle u_1, u_2\rangle \in st(T) \\
I_\varepsilon(\langle u_1, u_2\rangle, l) & \Rightarrow & I_\varepsilon(u_2, UR(u_1)) & \\
I_\varepsilon(u_1, l),\ \ I_{\varepsilon'}(u_2, l') & \Rightarrow & I_{\varepsilon+\varepsilon'}(\{u_1\}_{u_2}, E) & \mid\ l \neq D({u_2}^{-1}),\ \ \{u_1\}_{u_2} \in st(T) \\
I_\varepsilon(\{u_1\}_{u_2}, l), I_{\varepsilon'}({u_2}^{-1}, l') & \Rightarrow & I_{\varepsilon+\varepsilon'}(u_1, D({u_2}^{-1})) & \mid\ l \neq E, \\
I_\varepsilon(u_1, l),\ \ I_{\varepsilon'}(u_2, l'),\ \ I_{\varepsilon''}(u_3, l'') & \Rightarrow & I_{\varepsilon+\varepsilon'+\varepsilon''}(\{u_1\}_{u_2}^{u_3}, Ep) & \mid\ l \neq Dp({u_2}^{-1}, u_3),\ \ \{u_1\}_{u_2}^{u_3} \in st(T) \\
I_\varepsilon(\{u_1\}_{u_2}^{u_3}, l), I_{\varepsilon'}({u_2}^{-1}, l') & \Rightarrow & I_{\varepsilon+\varepsilon'}(u_1, Dp({u_2}^{-1}, u_3)) & \mid\ l \neq Ep, \\
I_\varepsilon(u, l), I_{\varepsilon'}(u, l') & \Rightarrow & Goal & \mid\ l \neq l',\ \ \varepsilon + \varepsilon' = 1, \\
I_\varepsilon(u, l),\ \ I_{\varepsilon'}(u^{-1}, l') & \Rightarrow & Goal & \ \ \ u \neq u^{-1},\ \ \varepsilon + \varepsilon' = 1 \\
Goal & \Rightarrow & &
\end{array} \right\}
$$

where ε, ε', $\varepsilon'' \in \{0, 1\}$ and $+$ denotes the Boolean operation or.

Figure 8. The set $\mathcal{S}'$ of ground Horn clauses (proof of Proposition 5)

Proof. First, we compute the set $\mathcal{G}_1$ of guessable symbols $g \in \mathcal{G}$ such that there exists a guessing–proof of $T \vdash g$ (Proposition 5). Then, we solve the existence of a DY–proof of $T \cup \mathcal{G}_1 \vdash s$ (Proposition 4). Both above steps can be performed in polynomial time in $\|T \cup \mathcal{G} \cup \{s\}\|_d$. □

Corollary 1. *Trace Insecurity (TI) is decidable in non–deterministic polynomial time when the intruder is passive.*

Proof. Let $\mathcal{P}$, (S_0, N_0), K_0, I and $s \in \mathcal{T}(\mathcal{F})$ be an instance of (TI). First, we have to choose which message (among those currently in the network) is read by the current protocol rule. It is the non–determinitic part of the algorithm. Then, we have to verify that the chosen message matches the message expected by the agents, and also that the secret s is deducible by the intruder by using all the messages previously gathered. These verifications can both be performed in polynomial time (Theorem 1). □

Corollary 2. *Protocol Insecurity (PI) is decidable in non–deterministic polynomial time when the intruder is passive.*

Proof. The complexity result follows from the facts that the length of every possible interleaving is polynomially bounded in the size of the problem, and we have an NP decision procedure for solving the trace insecurity problem. □

Note that the three above complexity results are independent from the concrete size of the dictionary. This is not surprising since we are interested in the complexity of the decision of the theoretical existence of a guessing attack rather than on the cost of mounting such an attack.

7. Decision Procedures in Presence of an Active Intruder

When the intruder is active, the situation is much more complicated than in Section 6. Indeed, in this case, the messages circulating over the network are either sent by the protocol participants (the processes) or by the intruder. Therefore, they cannot be computed in advance, and will be represented by terms with variables. More precisely, (TI) is reduced to a problem of symbolic constraint solving. An effective constraint solving procedure is presented in Section 8.

7.1. Symbolic Constraints

Definition 10. *(constraint, solution)*
A constraint is a sequent of the form $T \Vdash_{\mathsf{dy}} u$ *(*DY–constraint*) or* $T \Vdash_{\mathsf{g}} u$ *(*guess–constraint*) where* T *is a finite subset of* $\mathcal{T}(\mathcal{F}, \mathcal{X})$ *and* $u \in \mathcal{T}(\mathcal{F}, \mathcal{X})$*. A solution of a finite set (or* system*)* $\mathcal{C}$ *of constraints is a grounding substitution* σ *such that for every* $T \Vdash_{\mathsf{dy}} u \in \mathcal{C}$ *(resp. every* $T \Vdash_{\mathsf{g}} u \in \mathcal{C}$*) there exists a DY–proof (resp. guessing–proof) of* $T\sigma \vdash u\sigma$*.*

Note that we do not assume that the constraints of the set $\mathcal{C}$ are variable disjoint.

Definition 11. *(well–formed)*
A finite set $\mathcal{C}$ *of constraints is well–formed if its elements can be ordered as* $T_0 \Vdash_{x_0} r_0, \ldots, T_l \Vdash_{x_l} r_l$ *such that the following conditions holds:*

- $0 \in T_0$,
- *for all* $i \leq l$*,* x_i *is either* dy *or* g*,*
- *for all* $i < l$*,* $T_i \subseteq T_{i+1}$*,*
- *for all* $i \leq l$*, for all* $x \in vars(T_i)$*, there exists* $j < i$ *such that* $x \in vars(r_j)$*.*

7.2. Reduction of (TI) to Constraint Solving

Let π be an instance of (TI) made of a protocol $\mathcal{P}$, a runnable initial configuration (S_0, N_0), a set $K_0 \subset \mathcal{T}(\mathcal{F})$, an interleaving I of length ℓ and a secret term $s \in \mathcal{T}(\mathcal{F})$. We associate to this input a set $\mathcal{C}(\pi) = \{C_0, \ldots, C_\ell\}$ of DY– and guess–constraints the solvability of which is equivalent to the problem of protocol insecurity in presence of an active intruder. We construct in parallel the constraints of $\mathcal{C}(\pi)$ and the sequences $T_0, \ldots, T_\ell$ of their hypotheses sets. Let $T_0 = N_0 \cup K_0$. For each $k < \ell$,

if $I_k \in \mathcal{G}$, then $C_k := T_k \Vdash_{\mathsf{g}} g$ (the intruder can deduce g by guessing) and $T_{k+1} := T_k \cup \{g\}$ (he adds this value to his knowledge),

Otherwise I_k is an index representing a process, say (p, σ). Let j be the number of instances of this index in $I_0 I_1 \ldots I_{k-1}$ and let $\mathsf{recv}(r); \mathsf{send}(s)$ be the jth instruction pair of the program p. Then $C_k := T_k \Vdash_{\mathsf{dy}} r\sigma$ ($r\sigma$ can be received from the network) and $T_{k+1} := T_k \cup \{s\sigma\}$ ($s\sigma$ is sent to the network).

And finally, if $s \in \mathcal{G}$ then $C_\ell := T_\ell \Vdash_{\mathsf{g}} s$ and otherwise, $C_\ell := T_\ell \Vdash_{\mathsf{dy}} s$ (the secret s is revealed).

Example 5. *(Kerberos protocol)*
The attack described in Section 2.3 can be executed starting with a (runnable) initial configuration $(S_0, \emptyset)$ *where* S_0 *contains five processes,* (p_i, σ_i) *with* $i \in \{0, 1, 4, 7, 8\}$*, and with the initial intruder's knowledge* $K_0 = \{0, S, pub(S), A, B, M_1, M_2, Ci, Ti\}$.

The processes p_0*,* p_1 *correspond to the respective roles* A *and* S *of session* α *which were described in Example 2, the process* p_4 *to the role* S *of session* β *and the other processes* p_7 *and* p_8 *to the role* S *and* B *of session* γ*, with:*

$$\sigma_4 = \left\{ \begin{array}{rcl} x^4_K & \mapsto & K', \\ x^4_{Ts} & \mapsto & Ts', \\ x^4_S & \mapsto & S, \\ x^4_{pub(S)} & \mapsto & pub(S) \end{array} \right\} \quad \sigma_7 = \left\{ \begin{array}{rcl} x^7_K & \mapsto & K'', \\ x^7_{Ts} & \mapsto & Ts'', \\ x^7_S & \mapsto & S, \\ x^7_{pub(S)} & \mapsto & pub(S) \end{array} \right\}$$

$$\sigma_8 = \{x^8_{pw(B,S)} \mapsto pw(B,S)\}$$

where K'*,* Ts'*,* K''*,* Ts'' *are constants of* $\mathcal{F}_0$.

The interleaving I *describing the trace of the attack is (together with the corresponding steps with the notation of Section 2.3):*

$$\begin{array}{cccccc} 0 & 1 & 4 & pw(A,S) & 7 & 8 \\ \alpha.0 & \alpha.1 & \beta.1 & pw(A,S) & \gamma.1 & \end{array}$$

The steps $\beta.0$*,* $\gamma.0$ *and* $\gamma.2$ *do not occur in* I *since they are performed by the intruder (on behalf of* A*) in the attack. The step* $(8,0)$ *(reception of last message by* B*) has no corresponding label.*
The associated constraint system $\mathcal{C}$ *is described in Figure 9.*

Lemma 2. *For every instance* π *of (TI), the size of* $\mathcal{C}(\pi)$ *is polynomial in the size of* π.

Lemma 3. *For every instance* π *of (TI), the set of constraints* $\mathcal{C}(\pi)$ *is well–formed.*

Proof. By the construction of $\mathcal{C}(\pi)$ and the hypothesis that the initial configuration (S_0, N_0) in π is runnable (Definition 3). □

7.3. Main Results

We shall give in the next section a resolution procedure for the problem of satisfiability of a set of constraints, showing the following theorem.

0	$T_0 := S_0, \quad C_0 := T_0 \Vdash_{\mathsf{dy}} 0$	A receives 0
	$T_1 := T_0, \{A, B, N_1, N_2, Ca, \{Ta\}_{pw(A,S)}\}_{pub(S)}$	A sends his request to S
1	$C_1 := T_1 \Vdash_{\mathsf{dy}} \{x^1_A, x^1_B, x^1_{N_1}, x^1_{N_2}, x^1_{Ca}, \{x^1_{Ta}\}_{pw(x^1_A,S)}\}_{pub(S)}$	S receives the request
	$T_2 := T_1, \{x^1_{N_1}, K \oplus x^1_{N_2}\}_{pw(x^1_A,S)}, \{x^1_A, K, Ts\}_{pw(x^1_B,S)}$	S answers the request
4	$C_2 := T_2 \Vdash_{\mathsf{dy}} \{x^4_A, x^4_B, x^4_{N_1}, x^4_{N_2}, x^4_{Ca}, \{x^4_{Ta}\}_{pw(x^4_A,S)}\}_{pub(S)}$	S receives a second request
	$T_3 := T_2, \{x^4_{N_1}, K' \oplus x^4_{N_2}\}_{pw(x^4_A,S)}, \{x^4_A, K', Ts'\}_{pw(x^4_B,S)}$	S answers, I diverts S answer
$pw(A,S)$	$C_3 := T_3 \Vdash_{\mathsf{g}} pw(A,S)$	I guesses $pw(A,S)$ using S answer
	$T_4 := T_3, pw(A,S)$	$pw(A,S)$ is added to I's knowledge
7	$C_4 := T_4 \Vdash_{\mathsf{dy}} \{x^7_A, x^7_B, x^7_{N_1}, x^7_{N_2}, x^7_{Ca}, \{x^7_{Ta}\}_{pw(x^7_A,S)}\}_{pub(S)}$	S receives a third request
	$T_5 := T_4, \{x^7_{N_1}, K'' \oplus x^7_{N_2}\}_{pw(x^7_A,S)}, \{x^7_A, K'', Ts''\}_{pw(x^7_B,S)}$	S answers, I diverts S answer
8	$C_5 := T_5 \Vdash_{\mathsf{dy}} \{x^8_A, x^8_K, x^8_{Ts}\}_{pw(B,S)}$	B accepts I's message
	$T_6 := T_5, 0$	B answers 0
	$C_6 := T_6 \Vdash_{\mathsf{dy}} K''$	and the secret K'' is revealed

Figure 9. The constraint system of Example 5.

Theorem 2. *Given a finite well–formed set $\mathcal{C}$ of constraints and a set $\mathcal{G}$ of guessable symbols, the existence of a solution is decidable in non–deterministic polynomial time in $\mathcal{C}$ and $\mathcal{G}$.*

Corollary 3. *Trace Insecurity (TI) is NP–complete when the intruder is active.*

Proof. The NP part follows from the above polynomial construction of set of constraints associated to instances of (TI) (Lemmas 2 and 3) and from Theorem 2.

Concerning the NP–hardness, if we choose $\mathcal{G} = \emptyset$, then we fall into the trace insecurity problem in presence of a standard active attacker (without guessing abilities). This problem can be shown NP–hard by a reduction from 3–SAT. Indeed, the reduction given by (Rusinowitch and Turuani, 2001) can easily be adapted in order to obtain a protocol with only one process. It allows to remove the choice of the interleaving, and hence to obtain a reduction from 3–SAT to the trace insecurity problem. □

Corollary 4. *Protocol Insecurity (PI) is NP–complete when the intruder is active.*

Proof. The NP part follows from the facts that the length of every possible interleaving is polynomially bounded in the size of the problem, and we have an NP decision procedure for solving the trace insecurity problem.

Concerning the NP–hardness, if we choose $\mathcal{G} = \emptyset$, then we fall into the problem of (Rusinowitch and Turuani, 2001) which has been shown NP–hard. □

8. Constraint Solving Procedure

We present in this section a non deterministic polynomial time algorithm to decide the satisfiability of a well–formed set of DY– and guess–constraints. The idea is that if there exists a solution, then there exists a minimal one whose dag–size is polynomial in the size of the system. This fact has been shown in (Rusinowitch and Turuani, 2001) for the verification of protocols in the Dolev–Yao model and we can use it to treat DY–constraints. However the case of guess–constraints is much more difficult because some results which are obvious for DY–proofs are not true for DY'–proofs (see Section 4.4). Indeed, we have shown that we can always assume that a DY–proof is in normal form, and we often use this result, but the transformation rules to normalize a DY'–proof of $T/T' \vdash u$ must be used very carefully since we can lose weak hypotheses in T' when we apply them. Moreover, a guessing–proof ends with an instance of the rule Compare which is inherently difficult.

8.1. Algorithm

The following non–deterministic decision algorithm takes as input a finite well–formed set $\mathcal{C}$ of DY– and guess– constraints, and checks the existence of a solution. Let $\{x_1, \ldots, x_m\} = vars(\mathcal{C})$.

1. choose for each $i \leq m$ a term $t_i \in st(\mathcal{C})$, and let σ be a most general unifier (if any) of the equational problem $\mathcal{E} = \{x_1 \approx t_1, \ldots, x_m \approx t_m\}$,

2. if σ is ground, check whether σ is a solution of $\mathcal{C}$.

If the answer of 2 is positive for some choices of 1, answer *Yes.* Otherwise, answer *No.*

Example 6. *(Kerberos protocol)*
We consider the set of constraints $\mathcal{C}$ described in Figure 9. At the first step of the algorithm, we build the following equational problem $\mathcal{E}$:

$$\begin{array}{c}
x^1_A \approx A,\ \ x^1_B \approx B,\ \ x^1_{N_1} \approx N_1,\ \ x^1_{N_2} \approx N_2,\ \ x^1_{Ca} \approx Ca,\ \ x^1_{Ta} \approx Ta,\\
x^4_A \approx A,\ \ x^4_B \approx B,\ \ x^4_{N_1} \approx N_1,\ \ x^4_{N_2} \approx N_2,\ \ x^4_{Ca} \approx Ca,\ \ x^4_{Ta} \approx Ta,\\[1em]
x^7_A \approx A,\ \ x^7_B \approx B,\ \ x^7_{N_1} \approx M_1,\ \ x^7_{N_2} \approx M_2,\ \ x^7_{Ca} \approx Ci,\ \ x^7_{Ta} \approx Ti,\\
x^8_A \approx A,\ \ x^8_K \approx K'',\ \ x^8_{Ts} \approx Ts''.
\end{array}$$

Let σ be the most general unifier of $\mathcal{E}$. Now, we can check that σ is a solution of $\mathcal{C}$, i.e. there exists a DY–proof of $C_i\sigma$ for each $i \in \{0, 1, 2, 4, 5, 6\}$ and also a guessing–proof of $C_3\sigma$.

For instance, the required guessing–proof can be obtained by applying the rule Compare *to the two DY'–proofs below:*

$$\dfrac{\dfrac{\dfrac{\{N_1, K \oplus N_2\}_{pw(A,S)} \in T_3\sigma}{T_3\sigma/\emptyset \vdash' \{N_1, K \oplus N_2\}_{pw(A,S)}}\ (\mathsf{A}') \qquad \dfrac{pw(A,S) \in \mathcal{G}}{T_3\sigma/pw(A,S) \vdash' pw(A,S)}\ (\mathsf{G})}{T_3\sigma/pw(A,S) \vdash' \langle N_1, K \oplus N_2\rangle}\ (\mathsf{D}')}{T_3\sigma/pw(A,S) \vdash' N_1}\ (\mathsf{UL}')$$

$$\dfrac{\dfrac{\dfrac{\{N_1, K' \oplus N_2\}_{pw(A,S)} \in T_3\sigma}{T_3\sigma/\emptyset \vdash' \{N_1, K' \oplus N_2\}_{pw(A,S)}}\ (\mathsf{A}') \qquad \dfrac{pw(A,S) \in \mathcal{G}}{T_3\sigma/pw(A,S) \vdash' pw(A,S)}\ (\mathsf{G})}{T_3\sigma/pw(A,S) \vdash' \langle N_1, K' \oplus N_2\rangle}\ (\mathsf{D}')}{T_3\sigma/pw(A,S) \vdash' N_1}\ (\mathsf{UL}')$$

8.2. Complexity

By construction, $\|\mathcal{E}\| \leq |vars(\mathcal{C})|.M$, where M is the maximal size of a term in $\mathcal{C}$. At step 1, the mgu σ (represented as a dag) can be computed in polynomial time in $\|\mathcal{E}\|$, using syntactic transformation rules for solving unification problems, see *e.g.* (Jouannaud and Kirchner, 1991). The dag–size of the terms in the codomain of σ is polynomial in the size of $\mathcal{E}$, hence in the size of $\mathcal{C}$.

The test of step 2 consists in checking that each constraint of $\mathcal{C}$ is satisfied by σ. By construction of $\mathcal{C}$, and according to the above bound on the size of σ, the instance of any constraint of $\mathcal{C}$ by σ has a dag–size polynomial in the size of $\mathcal{C}$. The results of Propositions 4 and 5 yield polynomial procedures for checking the satisfaction of each constraint of $\mathcal{C}$ by σ. Altogether, the complexity of the algorithm is polynomial in the sizes of $\mathcal{G}$ and $\mathcal{C}$.

8.3. Completeness

The completeness of our algorithm is ensured by the corollary 5 of the key proposition 6. The following technical lemmas 4,5, 6 will be used in the proof of this proposition. Given a proof P of a sequent $T \vdash u$, the aim of Lemmas 4 and 6 is to ensure the existence of a particular proof of $T \vdash u$ which respects some extra conditions in order to guarantee some results when we are going to apply transformations, as replacement, on proof trees.

The following lemma 4 has been proved in (Rusinowitch and Turuani, 2001) for a slightly less general model than the one we have presented here, roughly, the DY–model without probabilistic encryption. Despite this slight difference in models, the proof of (Rusinowitch and Turuani, 2001) can be translated in a straightforward way to prove the following lemma.

Lemma 4. *Let P be a DY–proof of $T \vdash t$ and P' be a minimal DY–proof of $T \vdash \gamma$ ending with a composition rule (E, Ep or P). There exists a proof of $T \vdash t$ in which γ is never decomposed.*

One may observe that in a guessing–proof of $T \vdash g$, the only relevant information in the weak hypotheses T' of a node $T/T' \vdash t$ is whether T' contains g or not. Below, in order to simplify the notations, the weak hypotheses in DY'–proofs will be noted g^+, g^- or $\emptyset$: g^+ and g^- represent arbitrary subsets of $\mathcal{G}$ respectively containing g and not containing g. Although $\emptyset$ is a subcase of g^-, we shall still use this notation to emphasis that every DY'–proof of $T/\emptyset \vdash u$ is isomorphic a DY–proof of $T \vdash u$. Note that a set of guessing–proof defined with the g^+ and g^- notation can be represented by a unique guessing–proof, which real contents of weak hypotheses are deduced from the leaves.

Lemma 6 below is an analogous of Lemma 4 for guessing proofs. Its proof requires the following auxiliary Lemma 5 which gives us sufficient conditions to have a guessing–proof of a given sequent $T \vdash g$.

Lemma 5. *Let P_1 and P_2 be two normal DY'–proofs of respectively $T/\emptyset \vdash' t$ and $T/g^+ \vdash' t$. There exist two DY'–proofs P_1' and P_2', subtrees of P_1 and P_2 respectively and a guessing–proof of $T \vdash g$ whose two sons are P_1' and P_2'.*

Proof. We prove this result by induction on the proof P_2. If the proof P_2 is an instance of (G), then we can apply the rule $\mathsf{Compare}$ since the conditions (i), (ii) and (iv) are clearly verified, and (iii) also because the proof P_1 of $T/\emptyset \vdash' t$ can not end with an instance of (G).

If the last rule of P_2 is $(\mathsf{UL'})$ then the conditions (i), (ii) and (iv) of $\mathsf{Compare}$ are clearly met. Either the proof P_1 and P_2 verify the condition (iii), and we can apply the rule $\mathsf{Compare}$, or these proofs end respectively by the instances $(T/\emptyset \vdash' \langle t_1, t_2\rangle, T/\emptyset \vdash' t_1)$ and $(T/g^+ \vdash' \langle t_1, t_2\rangle, T/g^+ \vdash' t_1)$ of $(\mathsf{UL'})$. In such a case we apply the induction hypotheses on the subtrees of P_1 and P_2 which root are labeled with $T/\emptyset \vdash' \langle t_1, t_2\rangle$ and $T/g^+ \vdash' \langle t_1, t_2\rangle$ respectively. The other cases, ($\mathsf{UR'}$, $\mathsf{P'}$, $\mathsf{D'}$, $\mathsf{Dp'}$, $\mathsf{E'}$, $\mathsf{Ep'}$) are very similar. □

In the following Lemma 6, equivalent of Lemma 4 for the guessing–proofs, we impose an extra condition (condition b) to ensure that the replacement we are going to perform on a such guessing–proof does not lose the only relevant information in the weak hypothesis set.

Lemma 6. *Let P be a guessing–proof of $T \vdash g$ and P' a minimal DY'–proof of $T/\emptyset \vdash' \gamma$ ending with a composition rule ($\mathsf{E'}$, $\mathsf{Ep'}$ or $\mathsf{P'}$). There exists a guessing–proof of $T \vdash g$ in which:*
(a) γ is never decomposed,
(b) every instance $(s_1, s_2, T/T' \vdash' \gamma)$ (resp. $(s_1, s_2, s_3, T/T' \vdash' \gamma)$ of the composition rule ($\mathsf{P'}$, $\mathsf{E'}$) (resp. $\mathsf{Ep'}$) is such that $g \notin T'$.

Proof. (for sake of readability, a part of the proof has been moved in Appendix A)
We make an induction on the number of instances of rules in P which do not satisfy (a) or (b). First, we consider the case of an instance which does not satisfy the condition (a), and we distinguish two subcases, depending on whether a premise of the instance is labeled with $T/g^+ \vdash' \gamma$ or $T/g^- \vdash' \gamma$. Let $\gamma = \langle \gamma_1, \gamma_2\rangle$ (the other cases $\gamma = \{\gamma_1\}_{\gamma_2}$ and $\gamma = \{\gamma_1\}_{\gamma_2}^{\gamma_3}$ are similar).

Case (a^+)*:* Let $(T/g^+ \vdash' \gamma, s)$ be an instance of $(\mathsf{UL'})$ (the case $(\mathsf{UR'})$ is similar) in P which does not satisfy the condition (a), and let P_1 the subproof of P whose root is the above $T/g^+ \vdash' \gamma$. We can apply the induction hypothesis to the guessing–proof P'' of $T \vdash g$ whose direct subproofs are P_1 and P'

Case (a^-)*:* Let P_1 and P_2 be the two direct (DY'–) subproofs of P. Let $(T/g^- \vdash' \gamma, s)$ be an instance of $(\mathsf{UL'})$ (the case $(\mathsf{UR'})$ is similar) in P_i (say $i = 1$ for simplicity) which does not satisfy (a) and let P'_1 be the tree obtained from P_1 by replacing the subproof of P_1 whose root is the above $T/g^- \vdash' \gamma$ by P' (whose root is $T/\emptyset \vdash' \gamma$). This tree P'_1 is a DY'–proof since $\emptyset$ is a subcase of g^- in our notation. It is not normal, and hence the condition (ii) of $\mathsf{Compare}$ is not satisfied by P'_1. We normalize the proof P'_1 using simplification rules, and show by induction on the number of simplification steps (using Lemma 5) that we obtain a guessing–proof P'' to which we can apply the induction hypothesis. The simplification rules and the detailed induction proof can be found in Appendix A.

Case (b): Let $(s_1, s_2, T/g^+ \vdash' \gamma)$ be an instance of the composition rule $(\mathsf{P'})$ in P which does not satisfy (b). Hence, we have a normal DY'–proof P'_1 of $T/g^+ \vdash' \gamma$, and we can apply Lemma 5 to P' and P'_1 in order to obtain a guessing–proof P'' of $T \vdash g$ on which we apply the induction hypothesis. □

Proposition 6. *Let σ be a minimal (w.r.t. $\ll$) solution of $\mathcal{C}$. For all $x \in vars(\mathcal{C})$, there exists $t \in st(\mathcal{C}) \setminus vars(\mathcal{C})$ such that $t\sigma = x\sigma$.*

Proof. (the proofs of the facts can be found in Appendix A)
We reason by contradiction. Assume that there exists $x \in vars(\mathcal{C})$ such that for all $t \in \mathcal{T}(\mathcal{F}, \mathcal{X}) \setminus \mathcal{X}$ with $t\sigma = x\sigma$, we have $t \notin st(\mathcal{C})$. We will show that under this condition there exists a smaller solution σ' of $\mathcal{C}$.

Let $\mathcal{C} = \{C_1, \dots C_\ell\}$ and for each $i \leq \ell$, let r_i be the target of C_i and $C_i\sigma$ be the (ground) constraint obtained from C_i by instantiating all the terms in its hypotheses and target with σ.

Fact 1. *If $x\sigma \in st(s\sigma)$ for some hypothesis s of C_i $(i \leq \ell)$, then there exists $j < i$ such that $x\sigma \in st(r_j\sigma)$.*

Fact 1 allows us to define: $m = \min\{j \mid x\sigma \in st(r_j\sigma)\}$. Note that the constraint C_m is a DY–constraint of the form $S_m \Vdash_{\mathsf{dy}} r_m$. Otherwise, r_m would be a ground term, hence $x\sigma \in st(r_m\sigma) \subseteq st(\mathcal{C})$, a contradiction.

Fact 2. *There exists a minimal DY–proof of $S_m\sigma \vdash x\sigma$ ending with a composition rule* (E, Ep *or* P)*.*

Now, we let δ be the replacement $\{x\sigma \mapsto 0\}$ (0 is a special constant introduced in Section 3). We will show that $\sigma' := \sigma\delta$ is also a solution of $\mathcal{C}$, which is a contradiction since $\sigma' \ll \sigma$. For this purpose, we have to build a proof of each $C_i\sigma'$, $i \leq l$. For each $i < m$, $x\sigma \notin st(C_i\sigma)$, by definition of m and Fact 1. Hence, $(C_i\sigma)\delta = C_i\sigma = C_i\sigma'$, *i.e.* σ' is a solution of C_i.
Let us show that σ' is also a solution of C_i for each $i \geq m$. We may note first that $C_i(\sigma\delta) = (C_i\sigma)\delta$, because of the hypothesis that there does not exists $t \in st(C_i) \setminus \mathcal{X}$ such that $t\sigma = x\sigma$. So, we are going to show that there exists a (DY– or guessing–) proof of $(C_i\sigma)\delta$ for each $i \geq m$.

Case (1): C_i is a guess–constraint $S_i \Vdash_{\mathsf{g}} g$. There exists a guessing–proof of $S_i\sigma \vdash g$ and moreover, thanks to Fact 2, there exists a normal DY'–proof of $S_i\sigma/\emptyset \vdash' x\sigma$ ending with a composition rule. Thanks to Lemma 6, there exists a guessing–proof GP of $S_i\sigma \vdash g$ which verify the condition (a) and (b). We shall build from GP a guessing–proof of $(S_i\sigma)\delta \vdash g$. We replace first in GP every subtree ended by an instance $(s_1, s_2, S_i\sigma/g^- \vdash' x\sigma)$ of composition rule (E', Ep' or P') by the following "instance" of (A'): $(x\sigma \in S_i\sigma, S_i\sigma/g^- \vdash' x\sigma)$. Then we apply δ to every term of the tree obtained, getting GP'.

Fact 3. *GP' is a guessing–proof of $(S_i\sigma)\delta \vdash g$.*

It follows that σ' is a solution of C_i.

Case (2): C_i is a DY–constraint $S_i \Vdash_{\mathsf{dy}} r_i$. By hypothesis, there exists a DY–proof of $S_i\sigma \vdash r_i\sigma$, and thanks to Fact 2 and Lemma 4, there exists a DY–proof of $S_i\sigma \vdash r_i\sigma$ in which $x\sigma$ is never decomposed. We can build as in the previous case a DY–proof of $(S_i\sigma)\delta \vdash (r_i\sigma)\delta$. ☐

Corollary 5. *If $\mathcal{C}$ admits a solution then there exists an equational problem of the form $\mathcal{E} = \{x_1 \approx t_1, \ldots, x_n \approx t_n\}$, where $\{x_1, \ldots, x_n\} = vars(\mathcal{C})$ and $t_1, \ldots, t_n \in st(\mathcal{C})$ such that σ is the unique most general unifier of $\mathcal{E}$.*

Proof. Let σ be a minimal solution of $\mathcal{C}$. By Proposition 6, for each x_i, $i \leq n$, there exists $t_i \in st(\mathcal{C}) \setminus vars(\mathcal{C})$ such that $t_i\sigma = x_i\sigma$, *i.e.* σ is a solution of $\mathcal{E} = \{x_1 \approx t_1, \ldots, x_n \approx t_n\}$. We can permute the indexes of the variables $x_1, \ldots, x_n$ in order to have $\mathcal{E}$ in dag solved form, *i.e.* such that:

$$\text{for every } 1 \leq i < j \leq n,\ x_i \notin \mathit{vars}(t_j) \tag{1}$$

(the x_i are pairwise distinct and every $t_j \notin \mathcal{X}$ by construction). The opposite would mean that $\mathcal{E}$ has no solution, using the completeness results for the dag based syntactic unification procedure presented in (Jouannaud and Kirchner, 1991). Indeed, the only transformation rule of this procedure applicable to a system of the form of $\mathcal{E}$ is the "occur–check", and its application would mean that $\mathcal{E}$ has no solution.

Hence, see (Jouannaud and Kirchner, 1991), $\mathcal{E}$ has a unique most general unifier $\theta = \theta_1 \ldots \theta_n$ where each θ_i is $\{x_i \mapsto t_i\}$. Since for each $i \leq n$, $t_i \in st(\mathcal{C})$, and hence $\mathit{vars}(t_i) \in \{x_1, \ldots, x_n\}$, and by condition (1), θ is ground. It implies that $\theta = \sigma$. $\square$

9. Related Work

Lowe presents in (Lowe, 2004) a formal CSP model of an intruder able of mounting dictionary attacks and a procedure to detect such attacks, which is implemented as an extension of the framework based on the protocol compiler Casper and the model checker FDR. We believe that our model is compatible with the one of (Lowe, 2004), though the formalisms differ, and hence that our decidability and complexity results are also valid for this system.

An other implementation of dictionary attacks detection is presented in (Corin et al., 2003). The authors, like us, define the existence of dictionary attacks by the solvability of a system of constraints, but unlike us, they use only Dolev–Yao constraints (of the form $\Vdash_{\mathsf{dy}}$). Indeed, the guessing–constraints $T \Vdash_{\mathsf{g}} g$ are encoded in (Corin et al., 2003) into some DY–constraints and negation of DY–constraints. However, this gives a definition of attacks strictly coarser than our Definition. For instance, the tree of the second example of Section 4.5, which is not a guessing–proof, and which, in our opinion does not represent a real dictionary attack, is considered as a dictionary attack in (Corin et al., 2003).

In (Cohen, 2002), the tool TAPS for protocol verification is extended to deal with offline dictionary attacks. The problem treated is the verification of protocols for an unbounded number of sessions, and hence, it is not a decision procedure. The definition of dictionary attacks is closed to the one of (Lowe, 2004).

In (Corin et al., 2004), offline dictionary attacks are modeled in a general and powerful framework: the applied–pi calculus of (Abadi and

Fournet, 2001). To illustrate their definition, the authors analyze manually in some protocols w.r.t. resistance to dictionary attacks. Applied–pi calculus is an extension of pi–calculus in which the messages are terms built on an arbitrary signature and considered modulo an equational theory E. This theory characterize the properties of operators such as encryption and decryption (like the deduction rules defining the intruder deduction abilities in our paper). This calculus allows also special processes called active substitutions, of the form $\{x = t\}$, meaning that the term t has been sent and read by the intruder but its contents are unknown, and t can be refereed using the variable x. A frame is a parallel composition of active substitutions with name restriction, denoted $\nu\overline{n}.\{x_1 = t_1, \ldots, x_n = t_n\}$. Two closed (without free variable) frames $\nu\overline{n}.\sigma$ and $\nu\overline{n}.\sigma'$ are statically equivalent if for every pair of terms u, v which do not contains the names of $\overline{n}$, we have $u\sigma =_E v\sigma$ iff $u\sigma' =_E v\sigma'$. It means that the frames cannot be distinguished by any pair of terms. Offline dictionary attacks can be defined naturally in this settings: roughly, a weak secret g can be guessed from the terms $t_1, \ldots, t_n$ which have been read by the intruder iff $\nu\overline{n}.\sigma$ and $\nu\overline{n}.\sigma'$ are not statically equivalent, where $\sigma = \{x_1 = t_1, \ldots, x_n = t_n\}$, and σ' is obtained from σ by replacing g by a fresh nonce g' (the intruder is able to distinguish a good guess of g from a wrong one). This definition is of course more general than our. In particular, it is not restricted to a particular equational theory. We believe however that our definition is an instance of the one of (Corin et al., 2004), for the case where E contains axioms for symmetric, public–key and probabilistic encryption, and pairing. Indeed, the DY'–proofs can be represented by terms with explicit destructors. For instance, a DY'–proof P ending with an instance of (D') is represented by $d(t_1, t_2)$ where d is an (explicit) symbol for decryption and t_1, t_2 represent the two direct subproofs of P. With such a correspondence, the two terms u and v in the definition of (non–)static equivalence represent the two direct subproof of a guessing–proof. Therefore, we believe that our procedure can be seen as a decision procedure for a particular case of (Corin et al., 2004).

10. Conclusion

We have defined a formal model of intruder with both deduction abilities à la Dolev–Yao, extended with a representation of probabilistic encryption, and guessing abilities. The verification of the protocol insecurity in this intruder model is shown decidable (when the number of sessions is bounded) in non–deterministic polynomial time. One may note that our procedure is not restricted to one intruder's guess per attack.

This paper gives a new complexity result and does not investigate practical issues, as the opposed of (Lowe, 2004; Corin et al., 2003): the algorithm given in this paper is not intended to be implemented as it is, due to its highly non–deterministic nature. However, the NP–completeness result of this paper should not be interpreted as the intractibility of the problem. Indeed, in this field, several security problems that belong theoretically to NP or even harder classes have appeared to be tractable in practice. Hence, we think it would be possible to implement the procedure in a clever manner in order to apply this framework on real protocols.

In this paper, we focused on offline dictionary attacks. The problem of online dictionary attacks, where the intruder uses an online exchange of messages to verify each of his guesses, is claimed to be realistic in some situations (Ding and Horster, 1995), but its formalization needs quite a different model than the one presented here.

Another possible extension is to work on an generalization of our procedure for the definition of protocol resistance against offline dictionary attacks of (Corin et al., 2004). In (Abadi and Cortier, 2004), it is shown that static equivalence of closed frames is decidable in polynomial time when the equational theory is a subterm theory (set of equations $t = s$ where s is a subterm of t). A consequence is that protocol security against dictionary attacks, following the definition of (Corin et al., 2004), is decidable in subterms theories for a bounded number of agents in the case of a passive intruder. The extension of this result to the case of an active intruder remains a challenging open question. We believe that the techniques developed in this paper could help in that direction.

Acknowledgement

We are grateful to the anonymous referees whose many relevant comments helped us to improve the paper.

References

Abadi, M. and V. Cortier: 2004, 'Deciding Knowledge in Security Protocols under Equational Theories'. In: *Proc. of the 31st International Colloquium on Automata, Languages, and Programming (ICALP'04)*, Vol. 3142 of *LNCS*. Turku (Finland), pp. 46–58.

Abadi, M. and C. Fournet: 2001, 'Mobile Values, New Names, and Secure Communication'. In: *Proc. of the 28th ACM Symposium on Principles of Programming Languages (POPL'01)*. London, (England), pp. 104–115.

Amadio, R. and W. Charatonik: 2002, 'On Name Generation and Set-Based Analysis in the Dolev-Yao Model'. In: *Proc. of the 13th International Conference on Concurrency Theory (CONCUR'02)*, Vol. 2421 of *LNCS*. Brno (Czech Republic), pp. 499–514.

Amadio, R. and D. Lugiez: 2000, 'On the Reachability Problem in Cryptographic Protocols'. In: *Proc. of the 11th International Conference on Concurrency Theory (CONCUR'00)*, Vol. 1877 of *LNCS*. Pennsylvania (USA), pp. 380–394.

Bellare, M., D. Pointcheval, and P. Rogaway: 2000, 'Authenticated Key Exchange Secure against Dictionary Attacks'. In: *Proc. of Advances in Cryptology (EUROCRYPT'00)*, Vol. 1807 of *LNCS*. Bruges (Belgium), pp. 139–155.

Bellovin, S. M. and M. Merritt: 1992, 'Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks'. In: *Proc. of IEEE Symposium on Security and Privacy*. pp. 72–84.

Blanchet, B.: 2004, 'Automatic Proof of Strong Secrecy for Security Protocols'. In: *IEEE Symposium on Security and Privacy*. Oakland, California, pp. 86–100.

Chevalier, Y., R. Küsters, M. Rusinowitch, M. Turuani, and L. Vigneron: 2003, 'Deciding the Security of Protocols with Diffie-Hellman Exponentiation and Product in Exponents'. In: *Proc. of the 23rd Conference on Foundations of Software Technology and Theoretical Computer Science (FST&TCS'03)*, Vol. 2914 of *LNCS*. Mumbai (India), pp. 124–135.

Cohen, E.: 2002, 'Proving Cryptographic Protocols Safe from Guessing Attacks'. In: *Proc. Foundations of Computer Security (FCS'02)*. Copenhagen, (Denmark).

Comon, H. and V. Cortier: 2005, 'Tree Automata with One Memory, Set Constraints and Cryptographic Protocols'. *Theoretical Computer Science* **331**(1), 143–214.

Comon-Lundh, H. and V. Shmatikov: 2003, 'Intruder Deductions, Constraint Solving and Insecurity Decision in Presence of Exclusive or'. In: *Proc. of the 18th Annual IEEE Symposium on Logic in Computer Science (LICS'03)*. Ottawa (Canada), pp. 271–280.

Corin, R., J. Doumen, and S. Etalle: 2004, 'Analysing Password Protocol Security Against Off-line Dictionary Attacks'. In: *Proc. of the 2nd International Workshop on Security Issues with Petri Nets and other Computational Models (WISP'04)*. Bologna, (Italy).

Corin, R., S. Malladi, J. Alves-Foss, and S. Etalle: 2003, 'Guess what? Here is a new tool that finds some new guessing attacks'. In: *Proc. of the Workshop on Issues in the Theory of Security (WITS'03)*. Warsaw (Poland).

Delaune, S. and F. Jacquemard: 2004, 'A Theory of Dictionary Attacks and its Complexity'. In: *Proceedings of the 17th IEEE Computer Security Foundations Workshop (CSFW'04)*. Asilomar, Pacific Grove, California, USA, pp. 2–15.

Dershowitz, N.: 1987, 'Termination of rewriting'. *Journal of Symbolic Computation* **3**, 69–116.

Ding, Y. and P. Horster: 1995, 'Undetectable On-line Password Guessing Attacks'. *Operating Systems Review* **29**(4), 77–86.

Dolev, D. and A. Yao: 1983, 'On the Security of Public-Key Protocols'. *IEEE Transactions on Information Theory* **29**(2), 198–208.

Durgin, N., P. Lincoln, J. Mitchell, and A. Scedrov: 1999, 'Undecidability of bounded security protocols'. In: *Proc. of the Workshop on Formal Methods and Security Protocols (FMSP'99)*. Trento, (Italy).

Goldwasser, S. and S. Micali: 1984, 'Probabilistic Encryption'. *Journal of Computer and System Sciences* **28**(2), 270–299.

Gong, L.: 1995, 'Optimal Authentication Protocols Resistant to Password Guessing Attacks'. In: *Proc. of the 8th Computer Security Foundations Workshop (CSFW'95)*. Kenmare (Ireland).

Gong, L., T. M. A. Lomas, R. M. Needham, and J. H. Saltzer: 1993, 'Protecting Poorly Chosen Secrets from Guessing Attacks'. *IEEE Journal on Selected Areas in Communications* **11**(5), 648–656.

Jouannaud, J.-P. and C. Kirchner: 1991, 'Solving Equations in Abstract Algebras: A Rule-Based Survey of Unification'. In: *Computational Logic - Essays in Honor of Alan Robinson*. pp. 257–321.

Katz, J., R. Ostrovsky, and M. Yung: 2001, 'Efficient Password-Authenticated Key Exchange Using Human-Memorable Passwords'. In: *Proc. of Advances in Cryptology (EUROCRYPT'01)*, Vol. 2045 of *LNCS*. Innsbruck (Austria), pp. 475–494.

Lowe, G.: 2004, 'Analysing Protocol Subject to Guessing Attacks'. *Journal of Computer Security* **12**(1), 83–98.

McAllester, D. A.: 1993, 'Automatic Recognition of Tractability in Inference Relations'. *Journal of the ACM* **40**(2), 284–303.

Millen, J. and V. Shmatikov: 2001, 'Constraint Solving for Bounded-Process Cryptographic Protocol Analysis'. In: *Proc. of the 8th ACM Conference on Computer and Communications Security (CCS'01)*.

Rusinowitch, M. and M. Turuani: 2001, 'Protocol Insecurity with Finite Number of Sessions is NP-Complete'. In: *Proc. of the 14th Computer Security Foundations Workshop (CSFW'01)*. Cape Breton (Canada), pp. 174–190.

Steiner, J. G., B. C. Neuman, and J. I. Schiller: 1988, 'Kerberos: An Authentication Service for Open Network Systems'. In: *Proc. of USENIX Winter Conference*. pp. 191–202.

Thayer, F. J., J. C. Herzog, and J. D. Guttman: 1999, 'Strand Spaces: Proving Security Protocols Correct'. *Journal of Computer Security* **7**(2).

Tsudik, G. and E. V. Herreweghen: 1993, 'Some Remarks on Protecting Weak Keys and Poorly-Chosen Secrets from Guessing Attacks'. In: *Symposium on Reliable Distributed Systems*. Princeton, New Jersey, (USA), pp. 136–141.

Wu, T.: 1998, 'The Secure Remote Password Protocol'. In: *Proc. of Internet Society Symposium on Network and Distributed System Security*. San Diego, CA, (USA), pp. 97–111.

Appendix

A. Completeness of the Decision Procedure

Lemma 6. *Let P be a guessing–proof of $T \vdash g$ and P' a minimal DY'–proof of $T/\emptyset \vdash' \gamma$ ending with a composition rule (E', Ep' or P'). There exists a guessing–proof of $T \vdash g$ in which:*
(a) γ is never decomposed,
(b) every instance $(s_1, s_2, T/T' \vdash' \gamma)$ (resp. $(s_1, s_2, s_3, T/T' \vdash' \gamma)$ of the composition rule (P', E') (resp. Ep') is such that $g \notin T'$.

Proof. Case (a^-)
We assume (w.l.o.g.) that $\gamma = \langle \gamma_1, \gamma_2 \rangle$ and $(T/g^- \vdash' \gamma, s)$ be an instance of (UL') which does not satisfy (a). Let P_1 and P_2 be the two direct DY'–subproofs of P. Let P_1' be the tree obtained from P_1 by replacing the subproof of P_1 whose root is the above $T/g^- \vdash' \gamma$ by P'.

We have a DY'–proof P_1' of $T/\emptyset \vdash u$ (u is the verifier used in the guessing–proof P) which is clearly not in normal form. So, we are going to normalize P_1' in order to obtain a guessing–proof P'' of $T \vdash g$ on which we can apply the induction hypothesis. Let us normalize P_1' using the simplification rules of Figure 6. During this normalization, the following instances of the simplification rules may be applied:

1.
$$\dfrac{\dfrac{\dfrac{Q_1}{T/\emptyset \vdash' u_1} \qquad \dfrac{Q_2}{T/\emptyset \vdash' u_2}}{T/\emptyset \vdash' \langle u_1, u_2 \rangle}\,(\mathsf{P}')}{T/g^- \vdash' u_1}\,(\mathsf{UL}') \quad \rightarrow \quad \dfrac{Q_1}{T/\emptyset \vdash' u_1}$$

We have a similar rule with (UR') instead of (UL').

2.
$$\dfrac{\dfrac{\dfrac{Q_1}{T/\emptyset \vdash' u_1} \qquad \dfrac{Q_2}{T/\emptyset \vdash' u_2}}{T/\emptyset \vdash' \{u_1\}_{u_2}}\,(\mathsf{E}') \qquad \dfrac{Q_3}{T/g^\epsilon \vdash' u_2^{-1}}}{T/g^\epsilon \vdash' u_1}\,(\mathsf{D}') \quad \rightarrow \quad \dfrac{Q_1}{T/\emptyset \vdash' u_1}$$

with $\epsilon \in \{-, +\}$.

3.
$$\dfrac{\dfrac{\dfrac{Q_1}{T/\emptyset \vdash' u_1} \quad \dfrac{Q_2}{T/\emptyset \vdash' u_2} \quad \dfrac{Q_3}{T/\emptyset \vdash' u_3}}{T/\emptyset \vdash' \{u_1\}_{u_2}^{u_3}}\,(\mathsf{Ep'}) \qquad \dfrac{Q_4}{T/g^\epsilon \vdash' u_2^{-1}}}{T/g^\epsilon \vdash' u_1}\,(\mathsf{Dp'})$$
$$\rightarrow \quad \dfrac{Q_1}{T/\emptyset \vdash' u_1}$$

with $\epsilon \in \{-,+\}$.

We have also an additional rule for deterministic encryption.

4.
$$\dfrac{\dfrac{\dfrac{Q_1}{T/\emptyset \vdash' \{u_1\}_{u_2}} \quad \dfrac{Q_2}{T/\emptyset \vdash' u_2^{-1}}}{T/\emptyset \vdash' u_1}\,(\mathsf{D'}) \qquad \dfrac{Q_3}{T/g^\epsilon \vdash' u_2}}{T/g^\epsilon \vdash' \{u_1\}_{u_2}}\,(\mathsf{E'})$$
$$\rightarrow \quad \dfrac{Q_1}{T/\emptyset \vdash' \{u_1\}_{u_2}}$$

with $\epsilon \in \{-,+\}$.

We have also some additional rules concerning pairing. We have omitted them since they do not bring any trouble. They transform a DY'-proof into a DY'-proof and can be treated as the rewriting rule 1.

We can show by induction on the number of simplification steps that every simplification rules is applied to a tree Q whose root is a node of P_1 and such that exactly one of the son of Q is a subproof of P'. This explains that the only instances of the simplification rules which may be applied are such that the set of weak hypotheses in the nodes of the upper part of the left member are empty, *i.e.* those described above.

Remark 2. *Note first that with our hypotheses ($\gamma = \langle \gamma_1, \gamma_2 \rangle$ is decomposed by* (UL')*) the rule of case 1 must necessarily be applied first in the simplification sequence, and this removes one instance of rule which do not satisfy* (a)*.*

We may first observe among the instances described above, some of them (for example 1), transforms a DY'–proof into a DY'–proof. Indeed, the simplification by one of these rules replaces a DY'–subproof of $T/\emptyset \vdash' u$ or $T/g^- \vdash' u$ by a DY'–subproof of $T/\emptyset \vdash' u$. Hence, the iterated simplification of P_1' by these rules terminates with a normal DY'–proof N_1 which has the same target u as the initial proof P_1. So, the conditions (i), (ii) and (iv) to apply the rule Compare on the proofs N_1 and P_2 are clearly verified.

Either the iterated simplification of P_1' by the simplification rules has not modified its last instance of inference rule and in such a case N_1 and P_2 verify (iii) since P_1 and P_2 verify also this condition. We let P'' be the guessing–proof of $T \vdash g$ with sons N_1 and P_2.

Or the iterated simplification of P_1' has modified its last instance of inference rule, hence N_1 is a DY'–proof whose set of weak hypotheses is empty and we can apply Lemma 5. In such a case, we let P'' the guessing–proof provided by Lemma 5. Since N_1 contains at least one instance of rule which does not satisfy (a) or (b) less than P_1, by the Remark 2 above, we can apply the induction hypothesis to P''.

However the rules of cases 2, 3 and 4 with $\epsilon = +$ may not transform a DY'–proof into a DY'–proof, because they may replace a DY'–subproof of $T/g^+ \vdash' u$ by a DY'–subproof of $T/\emptyset \vdash' u$, hence the information about g is lost. Assume that one of these instances is applied during the simplification of P_1' and consider the first application.

If the first application case is 2 (with $\epsilon = +$), let Q_1' be the right DY'–proof above the instance of (D') (this is a DY'–proof of $T/g^+ \vdash' u_2^{-1}$) and Q_2' be the right DY'–proof above the instance of (E') (this is a DY'–proof of $T/\emptyset \vdash' u_2$).

By hypothesis, these subproof Q_1' of Q_2' are minimal hence normal. Hence, we have a normal DY'–proof of $T/g^+ \vdash' u_2^{-1}$ and a normal DY'–proof of $T/g^+ \vdash' u_2$. If $u_2^{-1} = u_2$ (it is a symmetric key) we can apply Lemma 5 as above. Otherwise, we can apply the rule Compare directly to these two normal DY'–proofs, since the condition (iii) is satisfied. By Remark 2, P'' contains at least one instance of rule which do not satisfy (a) less than P and we can apply the induction hypothesis to P'' to conclude.
The situation is almost the same for the cases 3 and 4 (with $\epsilon = +$). □

Fact 1. *If $x\sigma \in st(s\sigma)$ for some hypothesis s of C_i $(i \leq \ell)$, then there exists $j < i$ such that $x\sigma \in st(r_j\sigma)$.*

Proof. This is a consequence of Lemma 3 since, by hypothesis, if $x\sigma \in st(C_i\sigma)$ then $x\sigma \in st(y\sigma)$ for some $y \in vars(C_i)$ (otherwise there exists a $t \in st(C_i) \setminus \mathcal{X}$ such that $t\sigma = x\sigma$). □

Fact 2. *There exists a minimal DY–proof of $S_m\sigma \vdash x\sigma$ ending with a composition rule (E, Ep or P).*

Proof. By hypothesis, there exists a DY–proof P of $S_m\sigma \vdash r_m\sigma$ and by Lemma 1, we can assume that P is a minimal DY–proof of $S_m\sigma \vdash r_m\sigma$. If P contains a node labeled by $S_m\sigma \vdash x\sigma$, then it is the root of a minimal subproof as expected. This proof indeed ends with a composition rule: otherwise, by minimality of P, we would have an occurrence of $x\sigma$ as a subterm of $S_m\sigma$, which contradicts the definition of m by Fact 1.

We show now that P necessarily contains one node labeled by $S_m\sigma \vdash x\sigma$. Assume that P contains no such node. We will construct recursively a path in P, from the root up to one leaf, every node of which is labeled by $S_m\sigma \vdash u$ such that $x\sigma \in st(u)$, and we shall show in parallel that the existence of such a path conducts to a contradiction.

By definition of m, the condition $x\sigma \in st(r_m\sigma)$ is true for the root of P, which is labeled by $S_m\sigma \vdash r_m\sigma$. Assume that this condition is also true for each node of (a prefix of) a path labeled by $S_m\sigma \vdash u_0, \ldots,$ $S_m\sigma \vdash u_k$, with $u_0 = r_m\sigma$ and let us consider the sons of $s = S_m\sigma \vdash u_k$ is P:

– if s has 1 son s_1 and (s_1, s) is an instance of (A), then $u_k \in S_m\sigma$ and $x\sigma \in st(u_k)$ contradicts the definition of m, because of Fact 1.

– if s has 1 son s_1 and (s_1, s) is an instance of (UL) or (UR), then u_k is a subterm of the target of s_1, hence also $x\sigma$, and we let s_1 be the next node of the path.

– if s has 2 sons s_1, s_2 and (s_1, s_2, s) is an instance of (D), then u_k is a subterm of the target of s_1 or s_2 (say s_1), hence also $x\sigma$, and we let s_1 be the next node of the path.

– if s has 2 sons s_1, s_2 and (s_1, s_2, s) is an instance of (P) or (E). By hypothesis, we have $x\sigma \neq u_k$ since we have assume that P contains no such node. So, $x\sigma$ is a strict subterm of u_k and it is also a subterm of the target of one of s_1 and s_2 (say s_1). Hence, we can let s_1 be the next node of the path. ☐

Fact 3. *GP' is a guessing–proof of $(S_i\sigma)\delta \vdash g$.*

Proof. The tree GP' has been obtained from a guessing-proof GP of $S_i\sigma \vdash g$ which verify the condition (a) and (b), by:

– replacing first in GP every subtree ended by an instance $(s_1, s_2, S_i\sigma/g^- \vdash' x\sigma)$ of composition rule (E', Ep' or P') by the following "instance" of (A'): $(x\sigma \in S_i\sigma, S_i\sigma/g^- \vdash' x\sigma)$.

– applying δ to every term of the tree obtained, getting GP'.

Note that the tree GP'' obtained is not a proof (since $x\sigma$ not belongs to $S_i\sigma$). The weak hypotheses are left untouched by δ since $x\sigma \notin st(\mathcal{G})$. We can show that the tree GP' we have obtained is a guessing–proof of $(S_i\sigma)\delta \vdash g$. We show first that in the two sons GP'_1 and GP'_2 of GP', for every node labeled by s' and with n sons labeled respectively by s'_1, $\ldots$, s'_n $(s'_1, \ldots, s'_n, s')$ is an instance of a rule of Figure 4.

– if $n = 1$ and (s'_1, s') is an instance of (A') added by replacement in GP of an instance (s_1, s_2, s) of a composition rule in the construction of GP' above. By construction, we have $s' = s\delta = (S_i\sigma)\delta/g^- \vdash' 0$ and we have $0 \in S_0 \subseteq (S_i\sigma)\delta$. Hence, (s'_1, s') is an instance of (A').

– if $n = 1$ and we are not in the above case, we have $(s'_1, s') = (s_1\delta, s\delta)$ where (s_1, s) is an instance of (G, A',UL' or UR') contained in GP.

Case (G): let (s_1, s) be $(u \in \mathcal{G}, S_i\sigma/u \vdash' u)$. As shown above, $x\sigma \notin \mathcal{G}$, therefore $(s'_1, s') = (s_1\delta, s\delta)$ is also an instance of (G).

Case (A'): this case is immediate.

Case (UL'), (UR'): by construction, we have $(s'_1, s) = (s_1\delta, s\delta)$ where (s_1, s) is an instance of (UL') (the case of (UR') is similar). Let (s_1, s) be $(S_i\sigma/g^\varepsilon \vdash' \langle u_1, u_2\rangle, S_i\sigma/g^\varepsilon \vdash' u_1)$. By condition (a) on GP, $\langle u_1, u_2\rangle \neq x\sigma$. Hence $\langle u_1, u_2\rangle\delta = \langle u_1\delta, u_2\delta\rangle$. Therefore (s'_1, s') is an instance of (UL$'$).

– if $n = 2$ and we have $(s'_1, s'_2, s') = (s_1\delta, s_2\delta, s\delta)$ where (s_1, s_2, s) is an instance of a composition rule, say (P') (the case of (E') is similar), contained in GP. Let (s_1, s_2, s) be $(S_i\sigma/g^{\varepsilon_1} \vdash' u_1, S_i\sigma/g^{\varepsilon_2} \vdash' u_2, S_i\sigma/g^{\varepsilon_1\cup\varepsilon_2} \vdash' \langle u_1, u_2\rangle)$ (where $\varepsilon_1 \cup \varepsilon_2 = -$ iff $\varepsilon_1 = \varepsilon_2 = -$). By construction of GP', $\langle u_1, u_2\rangle \neq x\sigma$. Indeed, according to condition (b), all the instances of composition rules with $x\sigma$ as target of the root has been replaced in the construction of GP''. Therefore, $\langle u_1, u_2\rangle\delta = \langle u_1\delta, u_2\delta\rangle$. Thus, (s'_1, s'_2, s') is an instance of (P$'$).

– if $n = 2$ and we have $(s'_1, s'_2, s') = (s_1\delta, s_2\delta, s\delta)$ where (s_1, s_2, s) is an instance of (D') (the case of (Dp') is similar), contained in GP. Let (s_1, s_2, s) be $(S_i\sigma/g^{\varepsilon_1} \vdash' \{u_1\}_{u_2}, S_i\sigma/g^{\varepsilon_2} \vdash' u_2^{-1}, S_i\sigma/g^{\varepsilon_1\cup\varepsilon_2} \vdash' u_1)$. By condition (a) on GP, $\{u_1\}_{u_2} \neq x\sigma$. Hence, $\{u_1\}_{u_2}\delta = \{u_1\delta\}_{u_2\delta}$. Therefore (s'_1, s'_2, s') is an instance of (D').

– if $n = 3$ and we have $(s'_1, s'_2, s') = (s_1\delta, s_2\delta, s\delta)$ where (s_1, s_2, s) is an instance of (Ep') contained in GP. Let (s_1, s_2, s_3, s) be $(S_i\sigma/g^{\varepsilon_1} \vdash' u_1, S_i\sigma/g^{\varepsilon_2} \vdash' u_2, S_i\sigma/g^{\varepsilon_3} \vdash' u_3, S_i\sigma/g^{\varepsilon_1 \cup \varepsilon_2 \cup \varepsilon_3} \vdash' \{u_1\}^{u_3}_{u_2})$ (where $\varepsilon_1 \cup \varepsilon_2 \cup \varepsilon_3 = -$ iff $\varepsilon_1 = \varepsilon_2 = \varepsilon_3 = -$). By condition (a) on GP, $\{u_1\}^{u_3}_{u_2} \neq x\sigma$. Hence, $\{u_1\}^{u_3}_{u_2}\delta = \{u_1\delta\}^{u_3\delta}_{u_2\delta}$. Therefore (s'_1, s'_2, s'_3, s') is an instance of (Ep').

We have shown that GP'_1 and GP'_2 are both DY'–proofs. The condition (i) of the rule (Compare) of Figure 5 is ensured, and the other conditions (ii)–(iv), for GP, are preserved in the construction of GP'. So, we have a guessing–proof GP' of $(S_i\sigma)\delta \vdash g$. □

Address for Offprints: Stéphanie Delaune, LSV, ENS de Cachan & CNRS, 61 avenue du Président Wilson, 94235 CACHAN Cedex - France, *tel:* +33 1 47 40 75 32, *fax:* +33 1 47 40 75 21, delaune@lsv.ens-cachan.fr